SECURIT

AD-A239 215

ENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REF Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Unclassified/Unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) CU-CSSC-91-17	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Center for Space Structures & Controls (Univ of Colorado)	6b. OFFICE SYMBOL (If applicable)
	7a. NAME OF MONITORING ORGANIZATION Naval Research Laboratory
6c. ADDRESS (City, State, and ZIP Code) Campus Box 429 University of Colroado Boulder, CO 80309-0429	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Naval Research Laboratory	8b. OFFICE SYMBOL (If applicable)
	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code) Code 5130S Washington D. C., 20375	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

ELECTE
AUG 07 1991

11. TITLE (Include Security Classification)
Development of Improved Modeling and Analysis Techniques for Dynamics of Shell Structures
Final Report

12. PERSONAL AUTHOR(S)
K. C. Park and C. Farhat

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final Report	FROM 6/9/87 TO 6/8/90	July 24, 1991	118

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	modeling, shell structures, parallel computations, finite elements

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This report contains two related research thrusts: improved shell structural modeling for underwater acoustics and massively parallel computations of shell dynamics response . Improvements on the existing ANS shell elements have been made and implemented on a testbed software; the module was then delivered to NRL for their applications. In addition, the so-called frequency-window tailoring of finite element models has been developed so that very high-frequency components can be accurately modeled with relatively coarse finite element grids, thus resulting in about a factor of five to ten times larger element size than has been possible in conventional finite element modeling.

(Continued on other side)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	Unclassified/unlimited
22a. NAME OF RESPONSIBLE INDIVIDUAL K. C. Park	22b. TELEPHONE (Include Area Code) (303)492-6330 / 22c. OFFICE SYMBOL

DD Form 1473, JUN 86 Previous editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

This document has been approved for public release and sale; its distribution is unlimited.

BLOCK 19 (CONT'D)

Second, several parallel modules using the C* language have been developed to run large-scale shell dynamics problems on the Connection Machines. These include: decomposer which takes as input an arbitrary mesh description, and produces a set of finite element data structures that can be loaded within one generic CM2 chip; mapper that assigns each of the data structures produced by the decomposer to a well defined chip; residual evaluator that controls the direct calculation of element residuals; element library that includes various finite elements; and visualization kernel that greatly facilitates the understanding of the computed results.

CU-CSSC-91-17

CENTER FOR SPACE STRUCTURES AND CONTROLS

DEVELOPMENT OF IMPROVED MODELING AND ANALYSIS TECHNIQUES FOR DYNAMICS OF SHELL STRUCTURES

Final Report

by

K. C. Park and Charbel Farhat

July, 1991

COLLEGE OF ENGINEERING
UNIVERSITY OF COLORADO
CAMPUS BOX 429
BOULDER, COLORADO 80309

91-07000

91 8 05 164

Final Report

on

Development of Improved Modeling and Analysis Techniques
for
Dynamics of Shell Structures
(Contract No. N00014-87-K-2018)

K. C. Park and Charbel Farhat
Department of Aerospace Engineering Sciences and
Center for Space Structures and Control
University of Colorado,Campus Box 429
Boulder, Colorado 80309

Accesion For
NTIS CRA&I
DTIC
Unannounced
Justification
By
Distribution /
Availability Codes
Dist | Avail and / or Special
A-1

July 1991

Submitted to:

Naval Research Laboratory
Code 5130S
Washington D. C., 20375
Technical Monitor: L. Schuetz-Couchman

TABLE OF CONTENTS

SUMMARY

ENCLOSED PAPERS AND REPORTS

Alvin, K. F. and Park, K. C.
Frequency-Window Tailoring of Finite Element Models for Vibration and Acoustics Analysis
Center for Space Structures and Controls,
Report No. CU-CSSC-91-16, July 1991,
University of Colorado, Boulder, CO.
Also to appear in Computational Methods for Acoustics Analysis, ASME 1991 Winter Annual Meeting, Atlanta, GA

Park, K. C. and Jensen, D. D.
A Systematic Determination of Lumped and Improved Consistent Mass Matrices for Vibration Analysis,
Proc. the 30th Structures, Dynamics and Materials Conference, AIAA Paper No. 89-1335, April 3-5,1989, pp.1532-1540.

Farhat, C., Sobh, N. and Park, K. C.
Transient Finite Element Computations on 65,536 Processors: The Connection Machine,
International Journal on Numerical Methods in Engineering, **30(1)**, *27-55 (1990).*

Farhat, C., Sobh, N. and Park, K. C.
Dynamic Finite Element Simulations on the Connection Machine,
International Journal of High Speed Computing,
Vol. 1, No. 2, pp. 289-302 (1989)

TABLE OF CONTENTS

SUMMARY

ENCLOSED PAPERS AND REPORTS

Alvin, K. F. and Park, K. C.
Frequency-Window Tailoring of Finite Element Models for Vibration and Acoustics Analysis
Center for Space Structures and Controls,
Report No. CU-CSSC-91-16, July 1991,
University of Colorado, Boulder, CO.
Also to appear in Computational Methods for Acoustics Analysis, ASME 1991 Winter Annual Meeting, Atlanta, GA

Park, K. C. and Jensen, D. D.
A Systematic Determination of Lumped and Improved Consistent Mass Matrices for Vibration Analysis,
Proc. the 30th Structures, Dynamics and Materials Conference, AIAA Paper No. 89-1335,
April 3-5,1989, pp.1532-1540.

Farhat, C., Sobh, N. and Park, K. C.
Transient Finite Element Computations on 65,536 Processors: The Connection Machine,
International Journal on Numerical Methods in Engineering, 30(1), *27-55 (1990).*

Farhat, C., Sobh, N. and Park, K. C.
Dynamic Finite Element Simulations on the Connection Machine,
International Journal of High Speed Computing,
Vol. 1, No. 2, pp. 289-302 (1989)

Park, K. C., Pramono, E., Stanley, G. M. and Cabiness, H. A.
The ANS Shell Elements: Earlier Developments and Recent Improvements,
Analytical and Computational Models of Shells,
Noor, A. K. et al (eds.), CED -Vol. 3,
1989, ASME, New York, 217-240.

Stanley, G. M., Cabiness, H. and Park, K. C.
Revised ANS Shell Elements: Implementation and Numerical Evluations, Computational Mechanics '88,
S. N. Atluri and G. Yagawa (editors),
Vol. 1, Springer-Verlag, 1988, pp. 26.v.1-4.

SUMMARY

This is a final report on the research project supported by the Naval Research Laboratory under Grant N00014-87-K-2018, entitled *Development of Improved Modeling and Analysis Techniques for Dynamics of Shell Structures*, which covered the period of 09 June 1987 to 08 June 1990. The objectives of the research have been: 1) to develop modeling and computational techniques suitable for the dynamic analysis of naval shell structures, and 2) to investigate, implement and evaluate tools for concurrent processing of very large structural engineering problems on the Connection Machine.

I. Research Accomplishments

Task 1: Shell Structural Modeling Techniques

This task consists of two related efforts: 1) improvement of the ANS shell elements (References 1 through 4) to better capture the coupling effects of mambrane-bending, membrane-transverse shear, and bending-transverse shear phenomena; 2) modeling techniques for improving the modes and mode shapes of 'dry' shell structures, particularly for the intermediate frequency ranges.

As a results of the first effort, Aashell element software module was implemented and delivered to NRL and its theoretical aspects was documented in References 13 and 14. Specifically, the new version of the ANS shell elements pass the patch test and considerably streamlined, resulting in substantial computational efficiency.

Regarding the modeling of shell structures by the finite elements for accurate intermediate-frequency computations, our initial effort began with tailoring of the mass matrices as documented in Reference 7. Even though such mass-matrix tailoring gave rise to a significant improvement of low-frequency computations, it fell short of yielding any appreciable improvement on intermediate to high-frequency computations. This has led us to tailor not only the mass matrices but also a component-by-component tailoring of stiffness matrices. For example, the tailored stiffness matrix consists of the tailored membrane, tailored bending and tailored transverse shear stiffness matrices. The synthesis to realize such a tailoring was facilitated by the use of the symbolic analysis technique developed previously in References 8 through 11.

The so-called *frequency-window tailoring* of finite element models as applied to bars and beams (Reference 12) demonstrates that it can accurately obtain very high-frequency components with relatively coarse finite element grids, about a factor of five to ten times larger element size that has been possible in conventional finite element modeling. This improvement, *if proved to be the case for general shells*, can have a significant impact on the finite element modeling capability of structural acoustics problems in the future.

Task 2: Parallel Computations on the Connection Machine

This task has focused on the use of the Connection Machine as applied to the explicit transient analysis of 'dry' shell structures. Our experience has been documented in References 5 and 6. In addition, a framebuffer generated visualization of the transient analysis of a generic submarine structure was produced as a video tape and delivered to the NRL technical monitor. Specifically, our effort concentrated on the development of several modules using the C* language provided by the Thinking Machine Corporation. The modules developed so far include: *decomposer* which takes as input an arbitrary mesh description, and produces a set of finite element data structures that can be loaded within one generic CM2 chip; *mapper* that assigns each of the data structures produced by the decomposer to a well defined chip; *residual evaluator* that controls the direct calculation of element residuals. where "direct" means that no element stiffness matrices are evaluated; and *element library* that includes a 3D 2-node truss, a 3D 2-node Bernouilli beam, a 3D 2-node Timoshenko beam, a 3D 8-node brick, a 2D 4-node quadrilateral and a 4-node ANS shell element. These modules, when interfaced with *visualization kernel* that was developed under AFOSR and NSF grants greatly facilitates the understanding of the computed results.

Our experience so far indicates that this highly parallel processor can outperform vector supercomputers such as the CRAY family on explicit computations but not on implicit ones. Based on the observations obtained during the present study, the following is a summary of the key conclusions:

(1) The current CM2 processor memory size of 64 Kbits penalizes high order elements in the sense that only small VP (virtual processor) ratios can be achieved. Thus the current configuration favors simpler elements. (This restriction should disappear in future CM2 models which will have 1Mbit of memory per processor.)

(2) Mesh irregularities slow down the computation speed in various ways.

(3) The Data Vault is very effective at reducing I/O time.

(4) The Virtual Processor concept outperforms substructuring.

REFERENCES

1. Park, K. C. and Stanley, G. M., "A Curved C^0 Shell Element Based on Assumed Natural-Coordinate Strains," To appear in *Journal of Applied Mechanics*, 1986.

2. Stanley, G. M., *Continuum-based shell elements*, Ph.D. Thesis, Stanford University, 1985.

3. Stanley, G. M., Park, K. C. and Hughes, T. J. R., "Continuum-based Resultant Shell Elements," Finite Element Method for Plate and Shell Structures, Volume 1: Element Technology, ed. by Hughes, T. J. R. and Hinton, E., Pineridge Press International, Swansea, U. K., 1986, pp. 1-45.

4. Park, K. C., Stanley, G. M. and Cabiness, H., "A Family of C^0 Shell elements Based on Generalized Hrennikoff's Method and Assumed Natural-coordinate Strains," *Finite Element Methods for Nonlinear Problems*, P. G. Bergan(editor), Springer-Verlag, 1986, 265-282.

5. Park, K. C., Pramono, E., Stanley, G. M. and Cabiness, H. A., "The ANS Shell Elements: Earlier Developments and Recent Improvements," in *Analytical and Computational Models of Shells*, Noor, A. K. et al (eds.), CED -Vol. 3, 1989, ASME, New York, 217-240.

6. Stanley, G. M., Cabiness, H. and Park, K. C., "Revised ANS Shell Elements: Implementation and Numerical Evluations," *Computational Mechanics '88*, S. N. Atluri and G. Yagawa (editors), Vol. 1, Springer-Verlag, 1988, pp. 26.v.1-4.

7. Park, K. C. and Jensen, D. D., "A Systematic Determination of Lumped and Improved Consistent Mass Matrices for Vibration Analysis," *Proc. the 30th Structures, Dynamics and Materials Conference*, AIAA Paper No. 89-1335, April 3-5,1989.

8. Park, K. C. and Flaggs, D. L., "An Operational Procedure for the Symbolic Analysis of the Finite Element Method," *Comp. Meth. Appl. Mech. Engr.*, 42, (1984) 37-46.

9. Park, K. C. and Flaggs, D. L., "A Fourier Analysis of Spurious Mechanisms and Locking in the Finite Element Method," *Comp. Meth. Appl. Mech. Engr.*, 46, (1984) 65-81.

10. Park, K. C., "Symbolic Fourier Analysis Procedures for C^0 Finite Elements," in : *Innovative Methods for Nonlinear Analysis*, W. K. Liu, T. Belytschko and K. C. Park(editors), Pineridge Press, Swansea, (1984), 269-293.

11. Park, K. C. and Flaggs, D. L., "A Symbolic Fourier Synthesis of a One-Point Integrated Quadrilateral Plate Element," *Comp. Meth. Appl. Mech. Engr.*, 48, (1985) 203-236.

12. Alvin, K. F. and Park, K. C. "Frequency-Window Tailoring of Finite Element Models for Vibration and Acoustics Analysis," to appear in *Computational Methods for Acoustics Analysis*, ASME Symposia Series, 1991, Winter Annual Meeting, 1991.

13. Farhat, C., Sobh, N. and K. C. Park, "Transient Finite Element Computations on 65,536 Processors: The Connection Machine," *International Journal on Numerical Methods in Engineering*, 30(1), 27-55 (1990).

14. C. Farhat, N. Sobh and K. C. Park, "Dynamic Finite Element Simulations on the Connection Machine," *International Journal of High Speed Computing*, Vol. 1, No. 2, pp. 289-302 (1989)

CU-CSSC-91-16

CENTER FOR SPACE STRUCTURES AND CONTROLS

FREQUENCY-WINDOW TAILORING OF FINITE ELEMENT MODELS FOR VIBRATION AND ACOUSTICS ANALYSIS

by

K. F. Alvin and K. C. Park

July, 1991

COLLEGE OF ENGINEERING
UNIVERSITY OF COLORADO
CAMPUS BOX 429
BOULDER, COLORADO 80309

Frequency-Window Tailoring of Finite Element Models for Vibration and Acoustics Analysis

K. F. Alvin and K. C. Park
Department of Aerospace Engineering Sciences and
Center for Space Structures and Controls
University of Colorado
Campus Box 429
Boulder, Colorado 80309

January 1991

ABSTRACT

A *frequency-window tailoring technique* is proposed for improved finite element modeling of structures for frequencies and their mode shapes in the acoustic range. The technique is based on the tailoring of three element attributes: frequency-tailored mass matrix, enhancement of stiffness matrix by a weighted spectral decomposition of membrane, bending and transverse shear energy for a desired frequency range (window), and a discrete Fourier synthesis of the resulting elemental eigenproblem models. The proposed technique has been applied to the vibration problems of bars and beams, which illustrate the effectiveness of the technique over conventional finite element modeling techniques.

1.0 Introduction

The response accuracy of finite element methods applied to linear structural dynamics problems is a function of both the finite element spatial discretization and the time domain integration technique applied to the coupled ordinary differential equations of the discrete model. Traditional techniques in improving the spatial discretization obtained via the finite element method are dominated by the so-called *h*-refinements and *p*-refinements.

In the first approach, the element mathematical formulation is held fixed while the number of elements (and number of global variables) is increased to obtain the desired spatial accuracy. The *p*-refinement, in contrast, holds the number of elements constant while increasing the order of the displacement field interpolations within the element. This also leads to an increase in the number of variables, but alters the fundamental element behavior. For example, one might refine a simple one-element truss model (i.e. spring element) by introducing a mid-point node. With a *h*-refinement, we would change the model from one linear-displacement element to two linear elements which share the mid-point node. A *p*-refinement, on the other hand, would exploit the additional node to replace the linear element with a single three-node bar element employing quadratic interpolations of the internal displacement field.

A common limiting factor for both of these refinements is that, once the grid sizes and approximate interpolation functions are decided upon, the accuracy that can accrue from the resulting discrete model is fixed. Specifically, while the convergence of the low frequencies and their mode shapes is in general assured as the grids and/or interpolation order are increased, there has been lack of a systematic convergence measurement for frequencies and their mode shapes ranging from intermediate to acoustics components. Consequently, this lack of high-frequency convergence assessment has led to the belief that it is hopeless to capture with high accuracy an acoustic range of frequencies and their mode shapes by the finite element approach within feasible computational means.

The development of consistent mass discretization [1], however, has motivated a number of investigators to study the wave dispersion characteristics of various mass modeling procedures for finite element analysis [2-11]. These efforts hhave included assessments of mass lumping for both constant-strain and higher-order elements, and point out clearly how mass modeling, independent of mesh size and displacement interpolation, can significantly affect model accuracy. Park and Jensen [12] use wave dispersion analysis to provide a systematic relatiionship between lumped and consistent mass discretizations, and show how averaging or tailoring the mass lumping can significantly improve response accuracy at specific higher frequencies. This approach is not adequate, however, for obtaining highly accurate acoustic frequency ranges, and furthermore leads to non diagonal mass matrices, which are undesirable for simulation via explicit time integration methods on massively-parallel computers. Thus, there remains a need for finite element approximations which accurately capture acoustic components while ideally maintaining a diagonal mass coefficient for computational considerations.

The present paper can be viewed as an initial attempt to fill this void so that a method that can eventually lead to adequate finite element modeling of the acoustic range frequencies and their mode shapes. To this end, we retain the two conventional model improvement techniques, viz., the h and p-refinements, but not to their extreme. We introduce a third component, which first breaks down the elemental attributes, parameterizes those decomposed attributes, and then recombines them based on a discrete Fourier synthesis so that the discrete characteristic dispersion curves match, *for a specified range of frequencies*, as closely as possible to those of the continuum case, hence the name *frequency-window tailoring technique*. The elemental attributes usually consist of mass matrices of linear and quadratic interpolations, stiffness matrices of membrane, bending and transverse shear components of constant and linear strain interpolations. It will be subsequently shown that the present frequency-window tailoring technique yields dramatically improved vibration analysis performance beyond either of the traditional methods for the same mesh size, thus providing an adequate accuracy with an affordable mesh refinement. The rest of the paper is ogranized as follows.

Section 2 briefly reviews the discrete Fourier analysis technique [13,14] as applied to two-noded and three-noded bar elements. The discrete dispersion curves are then compared with that of the continuum case. The establishment of this comparison forms the basis for the present element synthesis or *frequency-window tailoring technique*. In addition, a similar discrete Fourier analysis of a Timoshenko beam and its correlation with the continuum case is presented. Thus, discretization accuracy of not only the membrane but also the bending and transverse shear phenomena can be assessed in a quantitive manner. Of particular interest from these analyses is the appearance of a pronounced jump in the frequency for the case of the quadratic bar element, and also for the case of the quadratic Timoshenko beam, although not as much pronounced. These jumps occur at the mode shape that corresponds to $k\ell = \pi/2$ where k is the wave number and ℓ is the elemental length, which is clearly at the admissible wave number range.

A parameterized tailoring of the bar element discretization is described in Section 3. For the case of the bar, a diagonal mass is first constructed as a linear combination of the linear and quadratic elements. A parameterized stiffness matrix is then constructed in a similar manner. These two parameters are then determined by requiring, for a specified range of frequencies, the discrete frequencies as close as possible to those of the continuum case. Encouraged by the success of the bar synthesis, a similar synthesis technique is applied to beam elements. This is carried out by introducing a three-parameter optimization process, viz., the mass parameter, the bending parameter and the transverse shear parameter. These results are reported in Section 4. Finally, discussions and some concluding remarks are offered in Section 5.

2.0 Discrete Fourier Analysis in Vibration of Finite Elements

The basic analysis tool that we are about to employ throughout the paper is the discrete Fourier method [12-14]. There are two important properties of the discrete Fourier method that are attractive for the present purposes: symbolic representation of the element attributes and the direct comparison of the discrete dispersion curves with the corresponding continuum ones. We now illustrate the method, for the sake of clarity and simplicity, by way of one-dimensional bar problems.

2.1 Fourier Analysis of Bar Elements

Consider the governing partial differential equation for a uniform elastic bar given by

$$\rho \frac{\partial^2 u}{\partial t^2} = E \frac{\partial^2 u}{\partial x^2} \tag{1}$$

where ρ is the mass density, E is Young's modulus, u is the axial displacement variable, and t and x are time and the axial position along the bar, respectively. The Fourier transformation of (1) can be performed by introducing the following form of u:

$$u = \hat{u} e^{i(\omega t - kx)} \quad . \tag{2}$$

which, when substituted into (1), yields its characteristic equation as

$$\left(\frac{\omega l}{c}\right)^2 = (kl)^2 \tag{3}$$

where l is a characteristic length, k is the wave number, c is the continuum wave speed equal to $\sqrt{E/\rho}$, and $\left(\frac{\omega l}{c}\right)$ and (kl) are the normalized (nondimensional) frequency and wave number, respectively. The characteristic equation (3) implies that, for the continuum case, the normalized frequency $\omega l/c$ is linearly proportional to the nondimensional wave number (kl).

Let us now consider a two-noded linear bar element. Assembling a uniform mesh of two bar elements of length l (see Figure 1), we obtain the discrete equation at the interior node m, as

$$\rho A l \ddot{u}_m = \frac{EA}{l} \left(u_{m-1} - 2u_m + u_{m+1}\right) \tag{4}$$

The discrete solution analogous to (2) is of the form

$$u_i(t) = \hat{u}_m e^{i(\omega t - k(x_i - x_m))} \tag{5}$$

which, when applied to (4), yields the following discrete characteristic equation:

$$\left(\frac{\omega l}{c}\right)^2 = (\bar{k}l)^2 \tag{6}$$

where the discrete wave number $\bar{k}$ is defined by

$$\bar{k} = \frac{\sin\frac{kl}{2}}{\frac{l}{2}}, \qquad 0 \le kl \le \pi \tag{7}$$

Hence, comparing (6) with (3) it is observed that the effect of discretization is embodied in the approximation of the continuum wave number k by the discrete wave number $\bar{k}$.

In order to extend the above analysis applicable to higher-order element interpolations, we generalize the discrete Fourier analysis as follows. Instead of representing the displacements of adjacent nodes by (5), the Fourier expansion is assumed to hold only for alternate nodes (see Figure 1), and so consider the two coupled difference equations at nodes $m-1$ and m:

$$\begin{aligned} \rho A l \ddot{u}_{m-1} &= \frac{EA}{l}(u_{m-2} - 2u_{m-1} + u_m) \\ \rho A l \ddot{u}_m &= \frac{EA}{l}(u_{m-1} - 2u_m + u_{m+1}) \end{aligned} \tag{8}$$

Substituting (5) into (8) together with the expressions

$$\begin{aligned} \ddot{u}_{m-1} &= -\omega^2 u_{m-1} & \ddot{u}_m &= -\omega^2 u_m \\ u_{m-2} &= e^{2ikl} u_m & u_{m+1} &= e^{-2ikl} u_{m-1} \end{aligned} \tag{9}$$

we obtain the following Fourier-transformed equation:

$$\mathbf{L}(k,\omega)\hat{\mathbf{u}} = \mathbf{0} \tag{10}$$

where

$$\mathbf{L}(k,\omega) = \begin{bmatrix} 2 - \left(\frac{\omega l}{c}\right)^2 & -\left(1 + e^{2ikl}\right) \\ -\left(1 + e^{-2} \quad\right) & 2 - \left(\frac{\omega l}{c}\right)^2 \end{bmatrix} \tag{11}$$

$$\hat{\mathbf{u}} = \{\, \hat{u}_{m-1} \quad \hat{u}_m \,\}^{\mathrm{T}} \tag{12}$$

The characteristic equation is found by requiring a non-trivial solution to (10), viz., $\det \mathbf{L} = 0$, from which two characteristic roots are found as

$$\left(\frac{\omega l}{c}\right)^2_1 = (\bar{k}l)^2$$
$$\left(\frac{\omega l}{c}\right)^2_2 = 4 - (\bar{k}l)^2 \tag{13}$$

The first root clearly agrees with that of the case of single interior-node equation (6), whereas the second does not appear to be consistent with the physics of the problem. On a closer examination, however, it can be shown that the second root corresponds nothing but to the π-phase-shifted case of (6). This can be explained as follows.

Note that, in terms of their eigenvectors, the first root is associated with $u_{m-1} = e^{ikl}u_m$, while the second with $u_{m-1} = e^{i(\pi-kl)}u_m$. Therefore, the proper wave number for the second root is

$$(kl)_2 = \pi - kl \quad , \qquad 0 \le kl \le \frac{\pi}{2} \tag{14}$$

so that

$$\left(\frac{\omega l}{c}\right)^2_2 = 4 - \left(\frac{\sin\frac{(\pi-kl)}{2}}{\frac{l}{2}} \cdot l\right)^2 = (\bar{k}l)^2 \tag{15}$$

The preceding analysis in terms of two coupled difference equations enables us to properly interprete the multiple characteristic roots associated with high-order elements. We are now in a position to take on the discrete Fourier analysis of quadratic bar elements.

The discretization of the bar equation (1) by the quadratic elements (see Figure 2) yields the following two coupled difference equations:

$$\rho A l \ddot{u}_{m-1} = \frac{EA}{l}(u_{m-2} - 2u_{m-1} + u_m)$$
$$\frac{\rho A l}{2}\ddot{u}_m = \frac{EA}{8l}(-u_{m-2} + 8u_{m-1} - 14u_m + 8u_{m+1} - u_{m+2}) \tag{16}$$

where the first is for the mid-node and the second for the end node and a diagonal mass matrix is used. The discrete Fourier-transformed operator $\mathbf{L}$ for the above 3-noded bar equations and the resulting characteristic equation can be derived as:

$$\mathbf{L}^Q(k,\omega) = \begin{bmatrix} 2 - \left(\frac{\omega l}{c}\right)^2 & -\left(1 + e^{2ikl}\right) \\ -\left(1 + e^{-2ikl}\right) & \frac{7}{4} + \frac{\cos 2kl}{4} - \frac{1}{2}\left(\frac{\omega l}{c}\right)^2 \end{bmatrix} \tag{17}$$

$$\left(\frac{\omega l}{c}\right)^4 - \frac{11 + \cos 2kl}{2}\left(\frac{\omega l}{c}\right)^2 + 3(1 - \cos 2kl) = 0 \tag{18}$$

Figure 3 compares the dispersion curves obtained for the linear and quadratic elements, as compared with that of the corresponding continuum equation (1). By invoking the same interpretation discussed in conjunction with the double roots given by (13) for the linear element, the upper root of the quadratic element is plotted versus the redefined wave number $\pi - kl$. It is clear from the results that the quadratic element performs better than its linear counterpart for an equivalent mesh size.

It is also noted, however, that the quadratic element gives rise to a discontinuity in the wave number/frequency relation at $kl = \pi/2$. That is, there is a range of frequencies the discrete model will simply "skip" over. As this discrete "forbidden" zone occurs in the middle of the spectrum of discrete behavior for the element mesh, it can be of particular concern and importance to acoustics and wave propagation analyses. Hughes [15], in studying mass matrix formulations and their effect on low frequency convergence for course meshes, produced numerical results consistent with the discrete Fourier analysis for quadratic bar and beam elements.

Figure 4 illustrates how dramatic this behavior can become for the solution of transient dynamics problems using quadratic bar elements. It shows the power spectral density of a nodal displacement response for a random initial displacement input, using a uniform mesh of 25 quadratic (3-node) bar elements and an implicit mid-point time integration algorithm. The "forbidden" discrete frequency zone in the range of frequences between 1.45 and 1.75 verifies numerically the Fourier analysis results for the quadratic bar element shown in Figure 3. Such forbidden discrete frequency zones, if they persist for quadratic elements such as 6 and 9-node shell elements, 10-node tetrahedral solids, and so forth, will have a profound effect on their ability to capture accurately high-frequency responses.

2.2 Discrete Fourier Analysis of Timoshenko Beam Elements

The strong form for the transverse beam vibration problem is the following set of coupled differential equations

$$\begin{aligned} \rho A \frac{\partial^2 w}{\partial t^2} &= GA\left(\frac{\partial^2 w}{\partial x^2} - \frac{\partial \theta}{\partial x}\right) \\ \rho I \frac{\partial^2 \hat{u}}{\partial t^2} &= EI\frac{\partial^2 \theta}{\partial x^2} - GA\left(\frac{\partial w}{\partial x} - \hat{u}\right) \end{aligned} \tag{19}$$

where A and I are the cross-sectional area and moment of inertia, G is the shear modulus, and w and θ are the transverse displacement and generalized rotation, respectively. The Fourier solution of (19) is assumed to be of the form

$$\hat{\mathbf{u}} = \hat{\mathbf{u}}_0 e^{i(\omega t - kx)} \tag{20}$$

where

$$\begin{aligned} \hat{\mathbf{u}} &= \lfloor w \quad \theta \rfloor^T \\ \hat{\mathbf{u}}_0 &= \lfloor w_0 \quad \theta_0 \rfloor^T \\ w_0 &= w(x=0, t=0) \\ \theta_0 &= \theta(x=0, t=0) \end{aligned} \tag{21}$$

leads to the transformed system

$$\mathbf{L}(\omega, k)\hat{\mathbf{u}}_0 = 0 \tag{22}$$

with the continuum Fourier matrix operator, $\mathbf{L}(\omega, k)$, given by

$$\mathbf{L}(\omega, k) = \begin{bmatrix} -\left(\frac{\omega l}{c}\right)^2 + \lambda(kl)^2 & -i\lambda l(kl) \\ i\frac{\lambda}{\gamma l}(kl) & -\left(\frac{\omega l}{c}\right)^2 + \frac{1}{\lambda}(kl)^2 + \frac{\lambda}{\gamma} \end{bmatrix} \tag{23}$$

where

$$c = \sqrt[4]{EG/\rho^2}, \quad \lambda = \sqrt{G/E}, \quad \gamma = I/Al^2 \tag{24}$$

and the continuum frequency equation derived from (23) is

$$\left(\frac{\omega l}{c}\right)^4 - \left[(\lambda + \frac{1}{\lambda})(kl)^2 + \frac{\lambda}{\gamma}\right]\left(\frac{\omega l}{c}\right)^2 + (kl)^4 = 0 \tag{25}$$

By inspection, there are two roots of (25) for each wave number value, and thus two unique vibration modes, the bending and the shear wave modes, which are plotted in Figure 5. The shear wave is associated with extremely high frequencies and not of primary concern

to the dynamist; hence we will focus on element tailoring only with respect to the bending mode.

For the case of the two-noded linear beam approximation, a similar procedure as employed for a linear bar element leads to the following discrete Fourier matrix operator, $\mathbf{L}(\omega, k)$, given by

$$\mathbf{L}(\omega,k) = \begin{bmatrix} -\left(\frac{\omega l}{c}\right)^2 + \lambda(\bar{k}l)^2 & -i\lambda l(\bar{k}l)\sqrt{\alpha^r} \\ i\frac{\lambda}{\gamma l}(\bar{k}l)\sqrt{\alpha^r} & -\left(\frac{\omega l}{c}\right)^2 + \frac{1}{\lambda}(\bar{k}l)^2 + \frac{\lambda}{\gamma}\alpha^r \end{bmatrix} \tag{26}$$

where $\bar{k}$ is defined by (7) and α^r is given by

$$\alpha^r = 1 - \frac{(\bar{k}l)^2}{4} \tag{27}$$

Therefore, the discrete frequency relationship for the linear Timoshenko beam element is given by

$$\left(\frac{\omega l}{c}\right)^4 - \left[(\lambda + \frac{1}{\lambda})(\bar{k}l)^2 + \frac{\lambda}{\gamma}\alpha^r\right] + (\bar{k}l)^4 = 0 \tag{28}$$

Figure 5 shows the resultant linear element frequency/wave number relationships obtained from (28), along with the corresponding continuum results from (25). Just as with the bar element, the linear beam element converges quite well at low frequencies, then gradually diverges from the continuum curve.

Similarly, for the case of a three-noded quadratic beam element, utilizing the transverse shear and bending stiffness and the lumped mass matrices, we obtain the discrete Fourier-transformed matrix operator $\mathbf{L}$ as follows

$$\mathbf{L}^Q = \tilde{\mathbf{K}}_{shear} + \tilde{\mathbf{K}}_{bending} - \left(\frac{\omega l}{c}\right)^2 \tilde{\mathbf{M}}_{lumped} \tag{29}$$

where

$$\tilde{\mathbf{K}}_{shear} = \begin{bmatrix} 2\lambda & 0 & -\lambda\left(1+e^{2ikl}\right) & \frac{\lambda l}{2}\left(1-e^{2ikl}\right) \\ 0 & \frac{2\lambda}{3\gamma} & -\frac{\lambda}{2\gamma l}\left(1-e^{2ikl}\right) & \frac{\lambda}{6\gamma}\left(1+e^{2ikl}\right) \\ -\lambda\left(1+e^{-2ikl}\right) & -\frac{\lambda l}{2}\left(1-e^{-2ikl}\right) & \lambda\left(\frac{7+\cos 2kl}{4}\right) & i\lambda l\left(\frac{\sin 2kl}{4}\right) \\ \frac{\lambda}{2\gamma l}\left(1-e^{-2ikl}\right) & \frac{\lambda}{6\gamma}\left(1+e^{-2ikl}\right) & -\frac{i\lambda}{\gamma l}\left(\frac{\sin 2kl}{4}\right) & \frac{\lambda}{6\gamma}(2-\cos 2kl) \end{bmatrix} \tag{30}$$

$$\tilde{\mathbf{K}}_{bending} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{2}{\lambda} & 0 & -\frac{1}{\lambda}\left(1+e^{2ikl}\right) \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{\lambda}\left(1+e^{-2ikl}\right) & 0 & \frac{1}{4\lambda}(7+\cos 2kl) \end{bmatrix} \tag{31}$$

$$\tilde{\mathrm{M}}_{lumped} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1/2 \end{bmatrix} \tag{32}$$

from which the following characteristic equation results:

$$\left(\frac{\omega l}{c}\right)^8 + c_1\left(\frac{\omega l}{c}\right)^6 + c_2\left(\frac{\omega l}{c}\right)^4 + c_3\left(\frac{\omega l}{c}\right)^2 + c_4 = 0 \tag{33}$$

where

$$\begin{aligned} c_1 &= \frac{\lambda}{3\gamma}(\cos 2kl - 4) - (\lambda + \frac{1}{\lambda})\left(\frac{11 + \cos 2kl}{2}\right) \\ c_2 &= 3(1 - \cos 2kl)(\lambda^2 + \frac{1}{\lambda^2}) + \frac{\lambda^2}{3\gamma^2}(1 - \cos 2kl) + \frac{1}{\gamma}(5 + \cos 2kl) \\ &\quad + \frac{\lambda^2}{12\gamma}(61 + 10\cos 2kl + \cos^2 2kl) + \frac{1}{4}(11 + \cos 2kl)^2 \\ c_3 &= \frac{3}{2}(\cos 2kl - 1)(11 + \cos 2kl)(\lambda + \frac{1}{\lambda}) + \frac{\lambda}{2\gamma}(\cos^2 2kl - 26\cos 2kl - 47) \\ c_4 &= 9(1 - \cos 2kl)^2 \end{aligned} \tag{34}$$

Figure 6 illustrates the characteristic frequency vs. wave number relation represented by (33), along with the corresponding continuum curves from (25). As with the quadratic bar vibration, the quadratic beam displays the frequency jumping phenomenon in both its bending and shear wave curves, although it is not as pronounced as in the case of membrane waves approximated by standard quadratic bar elements. At this point the analyst might conclude that, as the quadratic beam elements do not exhibit as pronounced forbidden discrete frequency zones as that found in the bar element, it may not be of concern for waves other than membrane cases. In the next section, however, we show that, while tailoring the element through parameterization can lead to overall improvements in performance, the analyst must be careful to avoid "over-optimizing" the higher frequency ranges as it may tend to accentuate the frequency-jumping phenomenon inherent in higher-order elements.

3.0 Frequency-Window Tailoring of Bar Elements

We have shown in the preceding section that the discrete Fourier analysis can, for a uniform mesh size, give an analytic/symbolic characterization of the finite element discretizations. For vibration analysis purposes, these characterizations can then be used to assess quantitatively the discretization accuracy as they can be directly compared with the corresponding continuum characteristics, viz., wave dispersion characteristics. Thus, the discrete Fourier analysis technique can be applied not only for the prediction of the resulting discretization for vibration analysis, but more importantly for the tailoring of element attributes in order to improve the finite element model accuracy. For example, it was shown in [12] that, for a given element stiffness matrix, a tailored mass matrix as a linear combination of the lumped and consistent mass matricies can significantly improve the accuracy of the low frequencies and their mode shapes. However, little improvement can be made for high-frequency components only by mass-matrix tailoring.

In order to improve the accuracy of the finite element models for the high-frequency components, we are motivated to tailor the element stiffness matrices for a given nodal pattern in addition to mass tailoring. A simple case to test such a stiffness tailoring concept is to employ a three-noded discrete nodal pattern for a bar so that one can work with two discrete element stiffness matrices: a 3×3 quadratic element stiffness matrix and the assembled stiffness matrix of two linear elements. Hence, we obtain the following three-noded tailored element mass and stiffness matrices:

$$\begin{aligned} \mathrm{M} &= \alpha_m \mathrm{M}^{quadratic} + (1-\alpha_m)\mathrm{M}^{linear} \\ \mathrm{K} &= \alpha_k \mathrm{K}^{quadratic} + (1-\alpha_k)\mathrm{K}^{linear} \end{aligned} \tag{35}$$

where M^{linear} and K^{linear} are the assembled matrices of two linear elements, and α_m and α_k are coefficients to be determined.

3.1 Discrete Fourier Analysis of Tailored Three-Noded Bar Element

The governing nodal difference equations at the mid-span node $m-1$ and the adjacent end node m are given by (see Figure 2)

$$\begin{aligned} \frac{\rho}{E}(3+\alpha_m)\ddot{u}_{m-1} &= (3+\alpha_k)\left(\frac{u_{m-2}-2u_{m-1}+u_m}{l^2}\right) \\ \frac{\rho}{E}(3-\alpha_m)\ddot{u}_m &= (3+\alpha_k)\left(\frac{u_{m-1}-2u_m+u_{m+1}}{l^2}\right)-\alpha_k\left(\frac{u_{m-2}-2u_m+u_{m+2}}{2l^2}\right) \end{aligned} \tag{36}$$

The Fourier-transformed discrete operator L for the tailored bar with the embedded performance parameters α_m and α_k can be derived as

$$\mathrm{L}(k,\omega)=\begin{bmatrix} 6+2\alpha_k-(3+\alpha_m)\left(\frac{\omega l}{c}\right)^2 & -(3+\alpha_k)\left(1+e^{2ikl}\right) \\ -(3+\alpha_k)\left(1+e^{-2ikl}\right) & 6+\alpha_k(1+\cos 2kl)-(3-\alpha_m)\left(\frac{\omega l}{c}\right)^2 \end{bmatrix} \tag{37}$$

The characteristic equation and its roots are then found to be

$$c_1\left(\frac{\omega l}{c}\right)^4-c_2\left(\frac{\omega l}{c}\right)^2+c_3=0 \tag{38}$$

where

$$\begin{aligned} c_1 &= 9-\alpha_m^2 \\ c_2 &= (3-\alpha_m)(6+2\alpha_k)+(3+\alpha_m)(6+\alpha_k(1+\cos 2kl)) \\ c_3 &= (6+2\alpha_k)(6+\alpha_k(1+\cos 2kl))-2(3+\alpha_k)^2(1+\cos 2kl) \end{aligned} \tag{39}$$

$$\begin{aligned} \left(\frac{\omega l}{c}\right)_1^2 &= \frac{c_2-\sqrt{c_2^2-4c_1c_3}}{2c_1} \\ \left(\frac{\omega l}{c}\right)_2^2 &= \frac{c_2+\sqrt{c_2^2-4c_1c_3}}{2c_1} \end{aligned} \tag{40}$$

3.2 Frequency-Window Tailoring of the Bar Element

Our objective in tailoring the bar element is to minimize the error between the exact and discrete eigenvalues in an average sense over a finite range of the frequency spectrum. To this end, for each frequency-window range $a \leq kl \leq b$, we perform the following minimization:

$$\min_{\alpha_m, \alpha_k} \int_a^b \left[1 - \frac{\left(\frac{\omega l}{c}\right)^2_{discrete}}{\left(\frac{\omega l}{c}\right)^2_{exact}} \right]^2 d(kl) \tag{41}$$

subject to (3), (39), and (40).

The optimization problem posed by (41) is not easily solvable in a closed-form sense, however, and since this element presents the simpliest form of Fourier equations that are of practical interest, we wish to consider a numerical complement to (41) which can be addressed by a standard quadratic optimzation technique. Therefore, the frequency-window tailoring method is recast as

$$\min_{\alpha_m, \alpha_k} J \tag{42}$$

where

$$J = \sum_{i=0}^{n} \left[1 - \frac{\left(\frac{\omega l}{c}\right)^2_{discrete}}{\left(\frac{\omega l}{c}\right)^2_{exact}} \right]^2_{kl = a + \frac{i}{n}(b-a)} \tag{43}$$

subject to (3), (39), and (40).

Numerical optimization of (42) was accomplished using quasi-Newton methods with central difference derivative approximations [16]. Both DFP and BFGS formulae were utilized in approximating the inverse Hessian matrix, though their performance was generally comparable.

3.3 Tailoring Results for the Bar Element

Twelve frequency windows were considered for tailoring of the bar element. The first six cover the wave number range of $0 \leq kl \leq \pi$ in even intervals; the results can be found in Table 1. The last six frequency windows cover progressively wider ranges of kl culminating in the final result which tailors α_m and α_k for the entire element spectral range; these results are shown in Table 2. In all cases, the numerical optimization used the quadratic element parameters as a starting point, and the initial and final values of J are noted. Figures 7 through 14 illustrate a selection of these results, along with the corresponding curves for the continuum equation and the discrete linear and quadratic elements. In all cases, the tailoring method produces marked improvements for each specified window, most dramatically for high-frequency and/or high-wave number ranges.

4.0 Frequency-Window Tailoring of Beam Elements

Encouraged by the improved accuracy of the tailored three-noded bar elements for window-by-window frequency ranges, this section extends the basic tailoring procedure to the Timoshenko beam element. In doing so, we adopt the same three-noded beam element nodal pattern as in the case of the tailored bar element with one important additional feature. The tailored element stiffness matrix consists of the tailored bending and the tailored transverse shear contributions, each of which in turn consists of a linear combination of a quadratic and a corresponding assembled two-element linear component, viz.,

$$\begin{aligned} \mathrm{K} &= \mathrm{K}_{shear} + \mathrm{K}_{bending} \\ \mathrm{K}_{shear} &= \alpha_s \mathrm{K}_{shear}^{quadratic} + (1-\alpha_s)\mathrm{K}_{shear}^{linear} \\ \mathrm{K}_{bending} &= \alpha_b \mathrm{K}_{bending}^{quadratic} + (1-\alpha_b)\mathrm{K}_{bending}^{linear} \\ \mathrm{M}_{lumped} &= \alpha_m \mathrm{M}_{lumped}^{quadratic} + (1-\alpha_m)\mathrm{M}_{lumped}^{linear} \end{aligned} \tag{44}$$

4.1 Tailoring Results for Beam Elements

The tailoring method proceeds as with the bar element by a discrete Fourier synthesis of the elemental eigenproblem, incorporating the resulting equations into the performance index (43), and a numerical optimization of the performance index to obtain the tailoring parameters α_s, α_b, and α_m. As noted in Section 2.2, the frequency-window tailoring has focused only on the bending curve waves. If necessary, a shear wave tailoring can be performed as well.

Results of several frequency-window tailoring can be found in Table 3 and Figures 11 through 14. Case I summarizes the tailored mass and stiffness parameters for the range of $\frac{\pi}{6} \leq kl \leq \frac{\pi}{3}$. As can be observed in Fig. 12, the accuracy improvement by the tailored element is rather dramatic. Errors of up to 6% in predicted frequencies from the nominal linear and quadratic elements are reduced to less than 0.5% over this entire low frequency window. In addition, Figure 11 illustrates how the tailored element also exhibits a marked improvement in frequency prediction over the full discrete frequency range, far beyond the designated "design" window.

Case II lists the tailoring of mass and stiffness matrices performed in the high-wave number range of $\frac{2\pi}{3} \leq kl \leq \frac{5\pi}{6}$. As Figure 13 illustrates, however, convergence within the desired spectrum does not imply superior overall behavior (the limits of the tailored spectrum are shown as vertical lines in the figure). Here, the optimized parameters tend to accentuate the forbidden discrete frequency zone, and allows significant errors in the low to middle spectrum that was optimized in Case I. These errors are as much as 60% with respect to the exact continuum solution in this area of the spectrum. Clearly, the influence of this "forbidden zone" is significant enough to warrant careful attention to the selection of frequency optimization windows and performance indicies.

In addition to the aforementioned problems in low frequency ranges, the unconstrained optimization of Case II resulted in parameters which lead to a negative-definite element stiffness matrix. Thus, the performance exhibited by the dispersion curves in Figure 13 is not practically realizable. Case IIa used the unconstrained optimization results as a starting point, then relaxed the stiffness tailoring parameters to regain the correct positive definite character of the element stiffness while retaining much of the improved element behavior in the desired frequency window. The issues of element feasibility and tailoring parameter constraints are covered in more detail in Section 4.2.

The final case studied, Case III, is an attempt to optimize the overall element behavior, producing reasonably small errors in the middle to high spectrum while maintaining accuracy in the low spectrum. The result (see Figure 14) shows that the magnitude of the forbidden zone has been reduced and the accuracy improved to within $\pm 10\%$ for the range of $0 \leq kl \leq \frac{2\pi}{3}$, while the errors of both the standard linear and quadratic elements are rapidly increasing as kl increases.

4.2 Tailoring Parameter Constraints

So far, we have used unconstrained optimization techniques to determine appropriate element tailoring parameters, by ignoring contraints that must be imposed on these variables to ensure element feasibility. Two important considerations are the preservation of element rigid-body modes and total mass. By using convex combinations of the linear and quadratic element formulations, it is easy to show that these are maintained for arbitrary values of the element performance parameters. In other words, these properties are preserved automatically. However, we must also preserve the positive definiteness of the mass matrix, and positive semi-definiteness of the stiffness matrix, and ensure the optimization does not introduce spurious element mechanisms.

For the bar element, these constraints result in the following parameter restrictions:

$$\begin{aligned} |\alpha_m| &< 3 \\ \alpha_k &> -3 \end{aligned} \tag{45}$$

Practically speaking, the restriction on the stiffness tailoring parameter is superfulous if the element is optimized over a reasonably broad range of its spectrum, and the Fourier analysis can be used a posteriori to examine the tailored element behavior, as it is a robust method for identifying deficiencies in the element formulation [17]. In other words, if we enforce the tailored mass matrix to be positive definite through an inequality constraint on α_m, and the Fourier analysis identifies the vibration behavior of the element as acceptable over its full spectral range, there is no need to constrain the stiffness tailoring parameters.

It should be noted, however, that while the Fourier analysis will identify problematic behavior in the element formulation, the evidence can be subtle and easily missed when examining the dispersion curve at a finite set of discrete wave number values. For example, in Section 4.1 it was noted that the optimization performed in Case II lead to a negative-definite element stiffness matrix. However, the negative eigenvalues found by examining the Fourier dispersion curve for the tailored element were in the range of $0 \leq kl \leq \frac{\pi}{10000}$, in other words in the lowest .01% of the element spectrum. A more reliable method (given the difficulty in deriving a general parameter constraint for elements more complex than bars) would to check the positive semi-definiteness of the element stiffness matrix at each step in the optimization process, and then limit the step size of the iteration if the semi-definiteness stiffness constraint is violated.

With this in mind, the beam element parameter constraint is given by

$$|\alpha_m| < 3 \tag{46}$$

By using a lumped mass formulation, it should be simple to determine the mass parameter constraint even in the case of complex two and three-dimensional elements. The computational overhead associated with checking the element stiffness matrix for negative

eigenvalues is very small compared to the basic optimization, and certainly much easier than deriving the dispersion curve at a sufficient number of values to ensure element feasibility.

4.3 Numerical Results for Tailored Finite Element Models

A final issue which must be addressed is the resultant accuracy of assembled finite element models using the tailoring methods described herein, both in terms of frequencies and mode shapes. The primary utility of the Fourier analysis within the context of the frequency-window tailoring technique is to closely correlate the temporal/spacial frequency characteristics of the parameterized element to the corresponding partial differential equations of motion. However, if the eigenvectors of an assembled model do not correlate well to the associated continuum wave shapes over the discrete spectrum, the model cannot be assumed to accurately capture the continuum wave dynamics. Fortunately, through numerical tests, it can be verified that the tailored elements maintain the mode shape characteristics of the linear and quadratic elements on which they are based.

For the bar vibration problem, with constrained ends, the wave shapes follow the Fourier spacial expansion used in the tailoring method. That is

$$\phi_n(x) = \sin \frac{n\pi x}{L} \qquad n = 1, 2, ... \tag{47}$$

Figure 15 demonstrates how the Fourier dispersion curve accurately predicts the resultant element's numerical behavior. The discrete points shown are the calculated frequencies of a 10-element mesh of tailored bars, with the given parameters, plotted against the corresponding wave number as determined by best-fitting the eigenvector of the mode to the Fourier spacial expansion. Not only is the appicability of the Fourier analysis verified, but the results also show how the discrete model eigenmodes uniformally cover the discrete spectrum, as is the case with linear bar elements. Figures 16 and 17 show that, for both moderate and high frequency modes, the eigenvectors of the tailored model retain the same basic character as their linear element counterparts, which themselves exactly match the continuum wave shapes at the node point locations. Figures 18 through 20 show similar results for a high-frequency-window tailoring case.

5.0 Discussions

A frequency-window tailoring technique is presented for improving finite element models that can be used to capture accurately frequencies and their mode shapes up to acoustic ranges. The tailoring of the element mass and stiffness matrices is achieved by a combination of linear and quadratic elements for both bar and beam elements. The tailoring parameters are optimized for each frequency window by minimizing the errors between the continuum and discrete characteristic dispersion curves. For the case of beam elements, the stiffness tailoring is further partitioned into a component-by-component contribution of the transverse shear and the bending stiffness matrices. Such a component-by-component tailoring has proved to be a key feature of the present tailoring technique.

The accuracy improvements realized by the present tailoring technique have been first predicted by the discrete Fourier analysis. The results demonstrate that the tailored elements dramatically improves the accuracy of both the frequencies and their mode shapes far beyond that of the linear or quadratic elements. This is also coroborated via numerical eigenvalue analyses. Although the results contained herein are primarily the analytical and numerical eigenvalues of the various element formulations, it should be remember that the fundamental product of the tailoring methods are the stiffness and mass interpolation parameters themselves, which are a function of the frequency window chosen by the analyst to optimize.

There are two overhead aspects associated with the present tailoring technique. The first is to employ a quadratic nodal topology for constructing the tailored element matrices. The second is to carry out several frequency window-by-window eigenvalue analyses in order to cove the entire range of frequencies of interest. These must be more than made up by a substantially reduced number of the nodal degrees of freedom by the present tailored elements. For example, for the wave-number range of $\frac{\pi}{3} \leq kl \leq \pi$, the tailored bar and beam elements yield the frequency accuracy within a few percent for $kl = 2.5$ from Fig. 13, whereas the conventional linear and quadratic beam elements must reduce their element sizes by a factor of about ten and five, respectively, if the same accuracy is to be maintained by these elements.

A straightforward extrapolation of the preceding accuracy comparison to two and three dimensional problems would result in 100 and 25 smaller nodal unknowns when the present technique is employed. As the extension of the present tailoring technique to plate, shell and solid elements appear to be straightforward, a bigger pay-off of the present technique may accrue as applied to two and three dimensional elements. For example, for shell elements, the tailoring of element stiffness can be achieved by synthesizing the membrane, the transverse shear, and the bending components. This is being carried out at present and we plan to report the results in a future occasion.

REFERENCES

1. Archer, J. S., "Consistent Mass Matrix for Distributed Mass Systems," *J. of the Structural Division*, Vol. 4, pp. 161-178 (1963).

2. Bazant, Z. P., "Spurious Reflection of Elastic Waves in Non-Uniform Finite Element Grids," *Computer Methods in Applied Mechanics and Engineering*, Vol. 16, pp. 91-100 (1978).

3. Bazant, Z. P., "Spurious Reflection of Elastic Waves in Non-Uniform Meshes of Constant and Linear Strain Elements," *Computers and Structures*, Vol. 15, No. 4, pp.451-459 (1982).

4. Belytschko, T. and Mullen, R., "On Dispersive Properties of Finite Element Solutions," *Modern Problems in Wave Propogation*, Edited by J. Miklowitz and J. D. Achenbach, John Wiley and Sons, New York, pp. 67-82 (1978).

5. Celep, Z. and Turhan, D., "Transient Wave Propogation in Constant and Linear Strain Elements," *J. Sound and Vibration*, Vol. 116, No. 1, pp. 15-23 (1987).

6. Fried, I. and Malkus, D. S., "Finite Element Mass Matrix Lumping by Numerical Integration with No Convergence Rate Loss," *Int. J. Solids Structures*, Vol. 11, pp. 461-466 (1975).

7. Hinton, E., Rock, T. and Zienkiewicz, O. C., "A Note on Mass Lumping and Related Processes in the Finite Element Method," *Earthquake Eng. and Structural Dynam.*, Vol. 4, pp. 245-249 (1976).

8. Krieg, R. D. and Key, S. W., "Transient Shell Response by Numerical Time Integration," *Int. J. Numerical Methods in Eng.*, Vol. 7, pp. 273-286 (1973).

9. Leckie, F. A. and Lindberg, G. M., "The Effect of Lumped Parameters on Beam Frequencies," *Aeronaut. Quarterly*, Vol. 14, pp. 224-240 (1963).

10. Tong, P., Pian, T. H. H., and Buciarelli, L. L., "Mode Shapes and Frequencies by the Finite Element Method Using Consistent and Lumped Mass," *J. Comp. Struct.*, Vol. 1, pp. 623-638 (1971).

11. Jiang, L. and Rogers, R. J., "Effects of Spacial Discretization on Dispersion and Spurious Oscillations in Elastic Wave Propogation," *Int. J. Numerical Methods in Eng.*, Vol. 29, pp. 1205-1218 (1990).

12. Park, K. C. and Jensen, D. J., "A Systematic Determination of Lumped and Improved Consistent Mass Matrices for Vibration Analysis", *AIAA Paper No. 89-1335*, April, 1989.

13. Park, K. C. and Flaggs, D. L., 1984, "An Operational Procedure for the Symbolic Analysis of the Finite Element Method", *Comp. Meth. Appl. Mech. Engr.*, 42, pp. 37-46.

14. Flaggs, D. L., *Symbolic Analysis of the Finite Element Method in Structural Mechanics*, Ph.D Thesis, Stanford University, 1988.

15. Hughes, T. J. R., *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*, Prentice-Hall, 1987.

16. Fletcher, R., *Practical Methods of Optimization*, 2nd Ed., Wiley-Interscience, 1987.

17. Park, K. C. and Flaggs, D. L., 1984, "A Fourier Analysis for Spurious Mechanisms and Locking in the Finite Element Method", *Comp. Meth. Appl. Mech. Engr.*, 46, pp. 68-81.

Case	Window	α_m	α_k	J_Q	J_{HP}
I	$0 \leq kl \leq \frac{\pi}{6}$	0.9990	1.0263	1.23×10^{-6}	0.10×10^{-6}
II	$\frac{\pi}{6} \leq kl \leq \frac{\pi}{3}$	0.4948	1.1840	8.44×10^{-4}	0.19×10^{-4}
III	$\frac{\pi}{3} \leq kl \leq \frac{\pi}{2}$	0.5409	1.3188	0.1018	4.60×10^{-4}
IV	$\frac{\pi}{2} \leq kl \leq \frac{2\pi}{3}$	0.6200	1.5631	0.1289	0.0041
V	$\frac{2\pi}{3} \leq kl \leq \frac{5\pi}{6}$	0.7439	1.9880	0.2729	0.0216
VI	$\frac{5\pi}{6} \leq kl \leq \pi$	0.9435	2.7517	1.4866	0.0851

Table 1: Tailoring Results for Bar Element (Narrow Frequency Windows)

Case	Window	α_m	α_k	J_Q	J_{HP}
VII	$0 \leq kl \leq \frac{\pi}{3}$	0.4948	1.1837	9.32×10^{-4}	0.25×10^{-4}
VIII	$\frac{\pi}{3} \leq kl \leq \frac{2\pi}{3}$	0.6539	1.4629	0.2136	0.0124
IX	$\frac{2\pi}{3} \leq kl \leq \pi$	0.8505	2.3723	1.7977	0.3571
X	$0 \leq kl \leq \frac{\pi}{2}$	0.5577	1.2887	0.1385	9.85×10^{-4}
XI	$\frac{\pi}{2} \leq kl \leq \pi$	0.8005	2.1897	1.9782	0.6210
XII	$0 \leq kl \leq \pi$	0.7804	2.1101	2.0701	0.7615

Table 2: Tailoring Results for Bar Element (Broad Frequency Windows)

Case	α_m	α_k	α_b	J_Q	J_{HP}
I $\frac{\pi}{6} \leq kl \leq \frac{\pi}{3}$	0.6692	1.0018	0.6244	0.0048	5.1801×10^{-5}
II $\frac{2\pi}{3} \leq kl \leq \frac{5\pi}{6}$	1.5321	1.0854	-0.2374	32.7325	1.1998×10^{-4}
IIa $\frac{2\pi}{3} \leq kl \leq \frac{5\pi}{6}$	1.5321	1.0800	0.5000	32.7325	0.1745
III $0 \leq kl \leq \frac{2\pi}{3}$	0.8508	1.0331	0.6685	0.7713	0.0653

Table 3: Tailoring Results for Beam Element

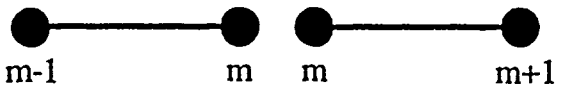


Figure 1: Nodal Geometry for Linear 2-node Line Elements

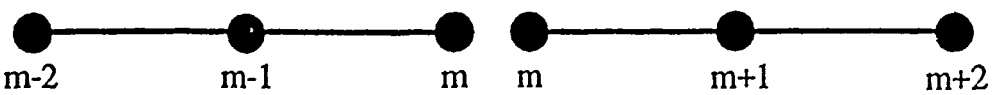


Figure 2: Nodal Geometry for Quadratic 3-node Line Elements

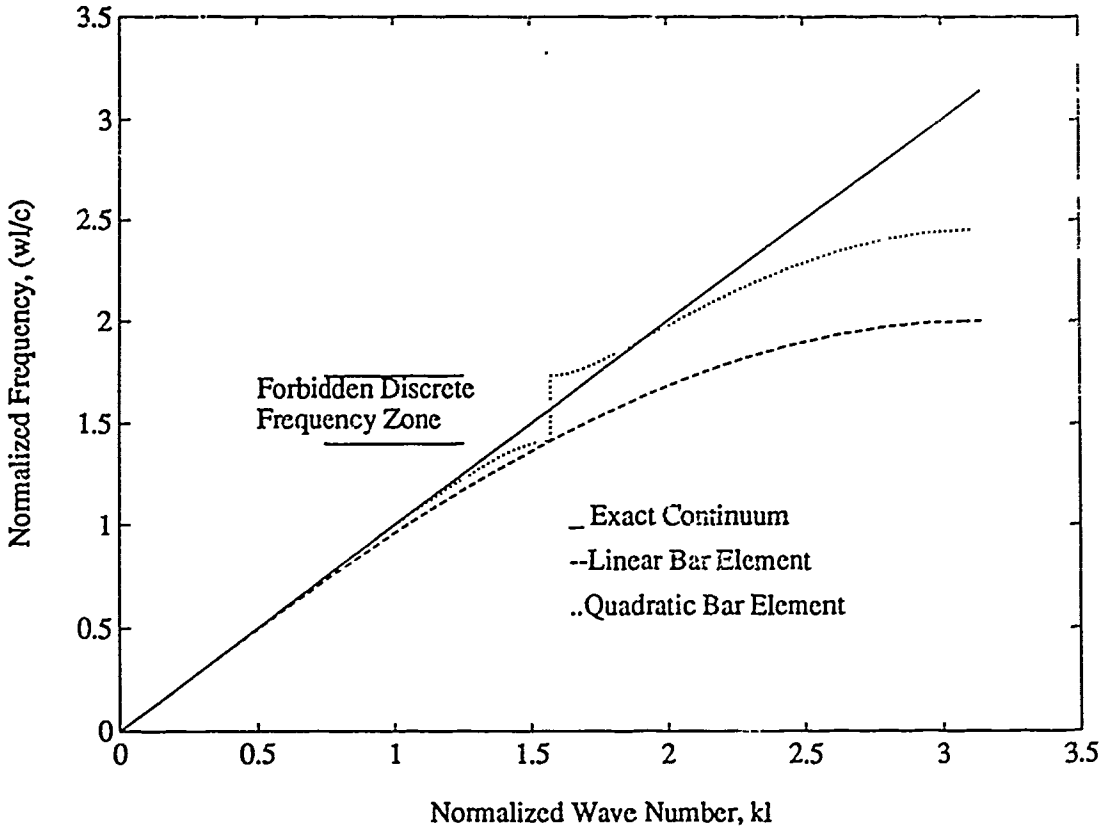


Figure 3: Fourier Analysis Results for Bar Elements

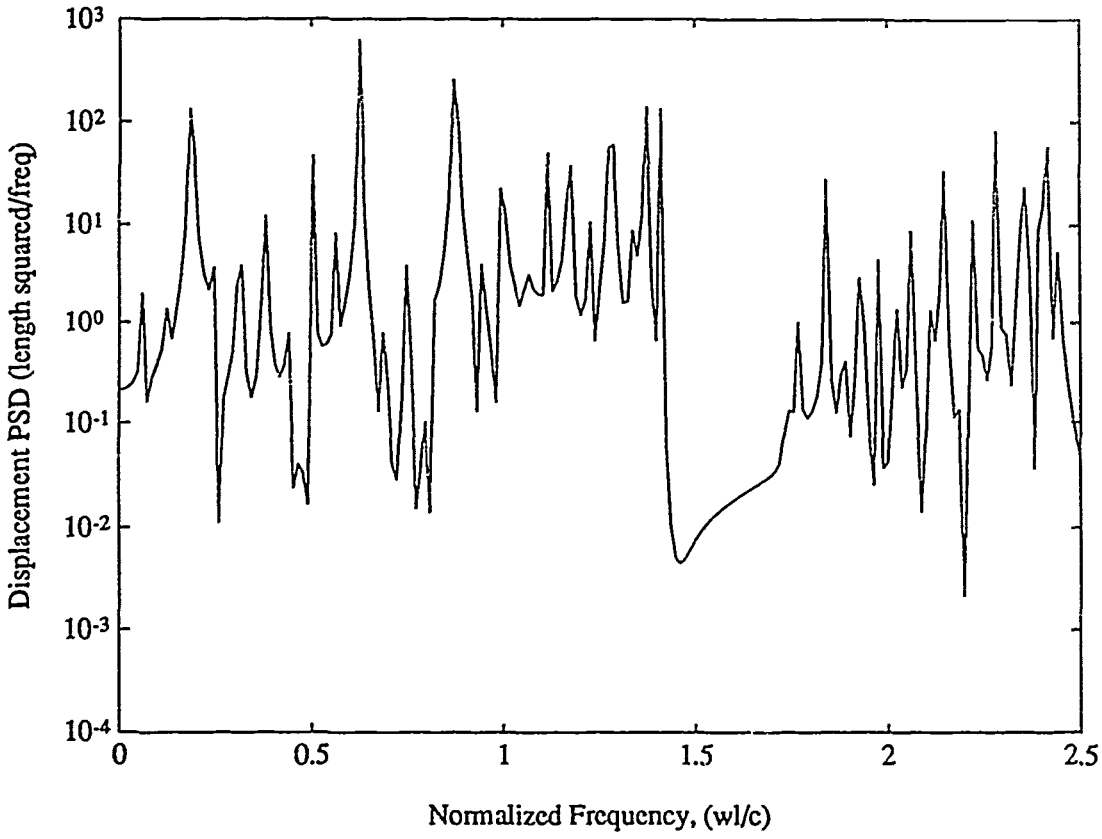


Figure 4: Effect of Quadratic Element "Forbidden Zone"
in Simulation of Bar Dynamics

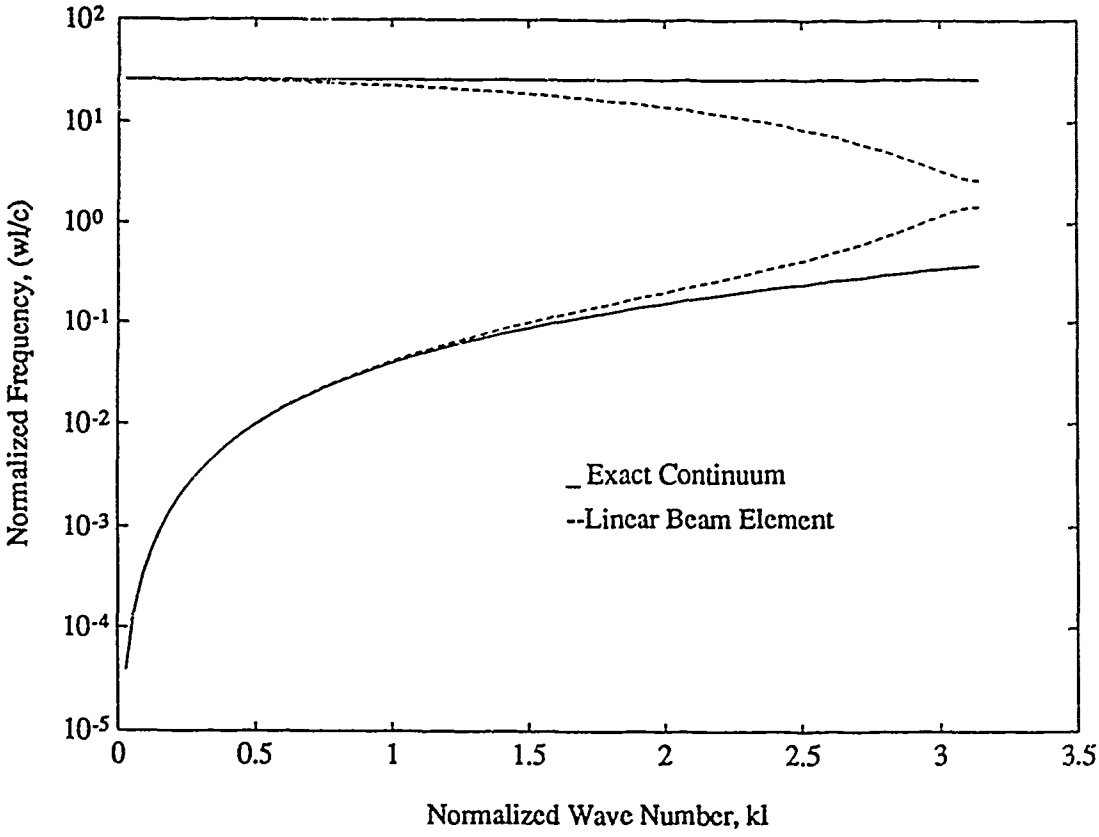


Figure 5: Fourier Analysis Results for Linear Beam Element

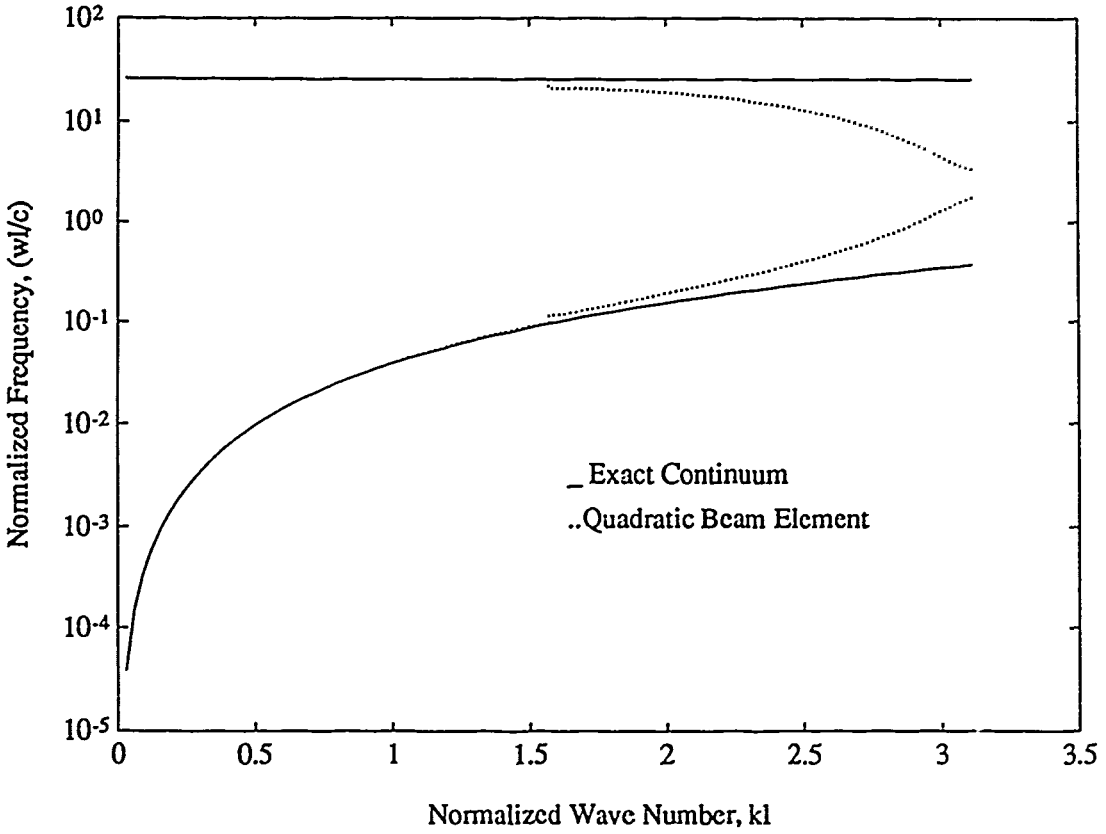


Figure 6: Fourier Analysis Results for Quadratic Beam Element

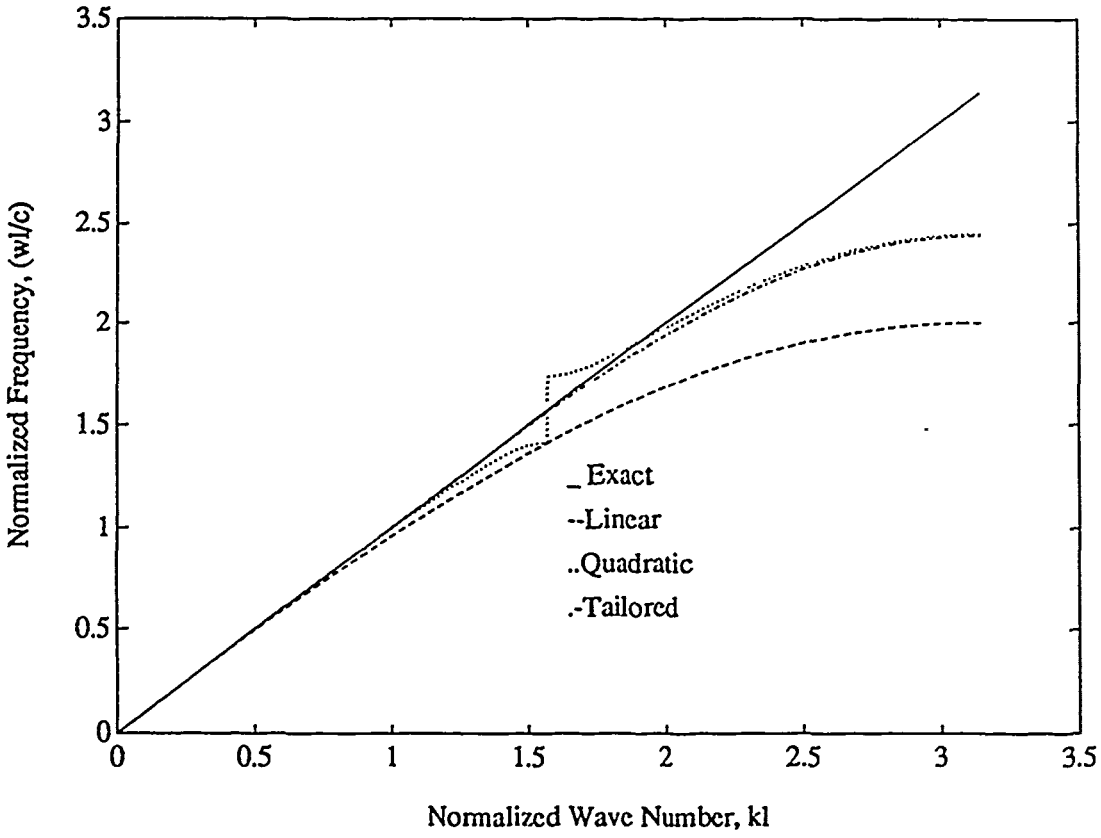


Figure 7: Fourier Analysis, Tailored Bar Element, Case III
Tailored Wave Range: $\frac{\pi}{3} \leq kl \leq \frac{\pi}{2}$
Parameters: $\alpha_m = 0.5409$, $\alpha_s = 1.3188$

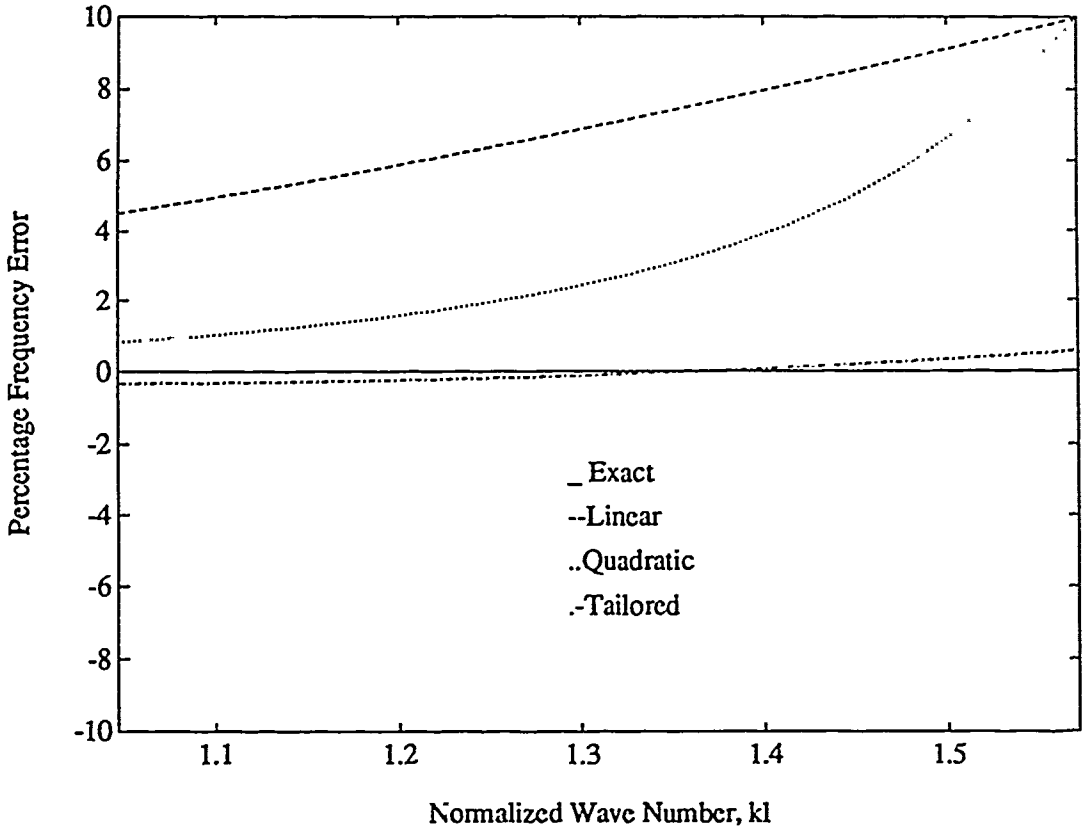


Figure 8: Frequency Error, Tailored Bar Element, Case III
Tailored Wave Range: $\frac{\pi}{3} \leq kl \leq \frac{\pi}{2}$
Parameters: $\alpha_m = 0.5409$, $\alpha_s = 1.3188$

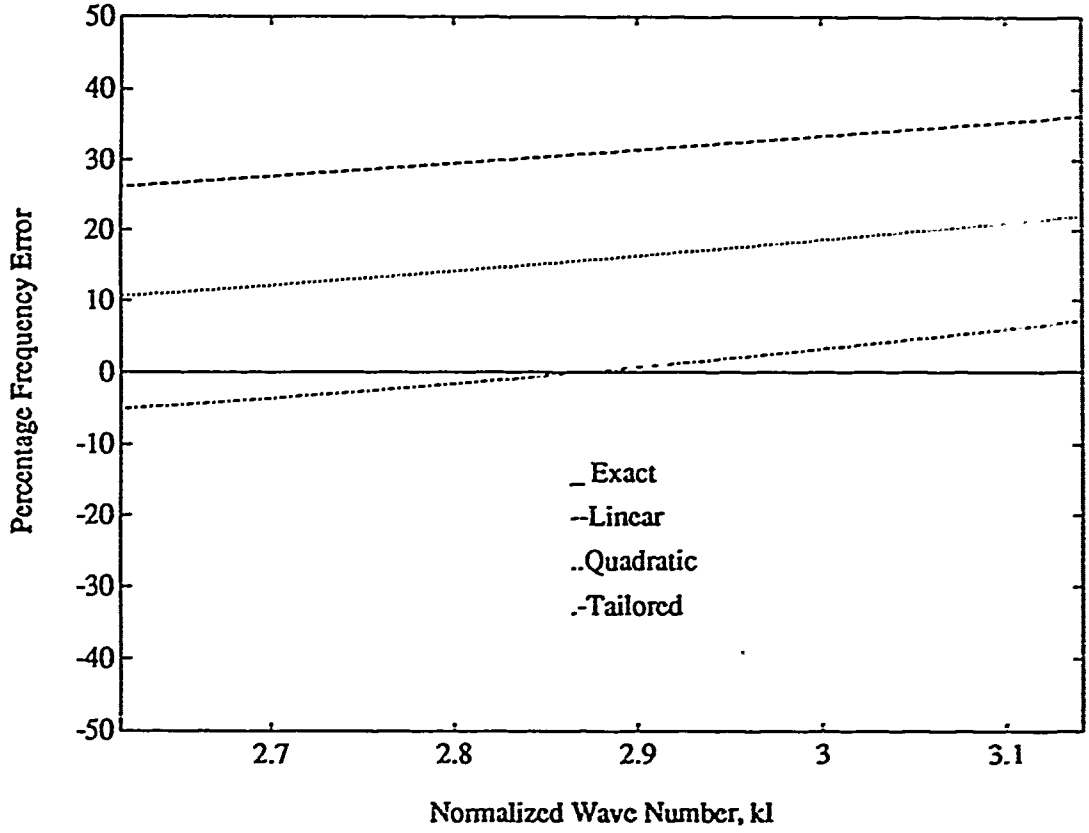


Figure 9: Frequency Error, Tailored Bar Element, Case VI
Tailored Wave Range: $\frac{5\pi}{6} \leq kl \leq \pi$
Parameters: $\alpha_m = 0.9435$, $\alpha_k = 2.7517$

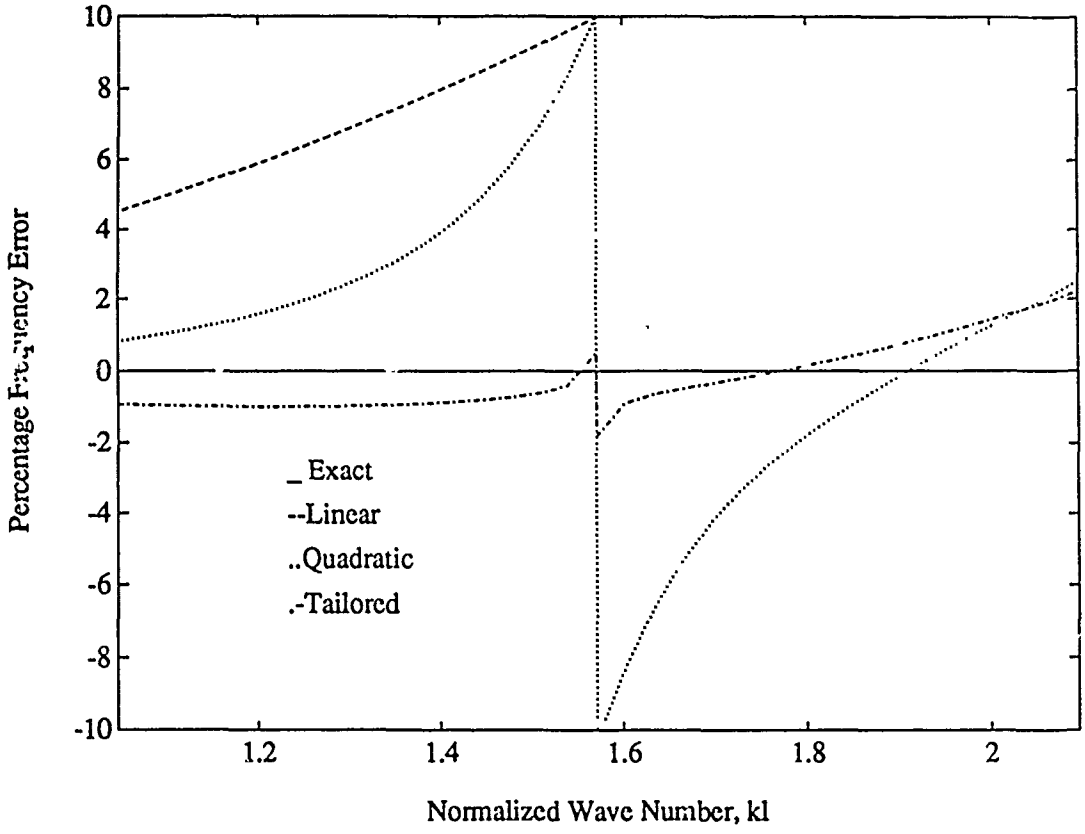


Figure 10: Frequency Error, Tailored Bar Element, Case VIII
Tailored Wave Range: $\frac{\pi}{3} \leq kl \leq \frac{2\pi}{3}$
Parameters: $\alpha_m = 0.6539$, $\alpha_k = 1.4629$

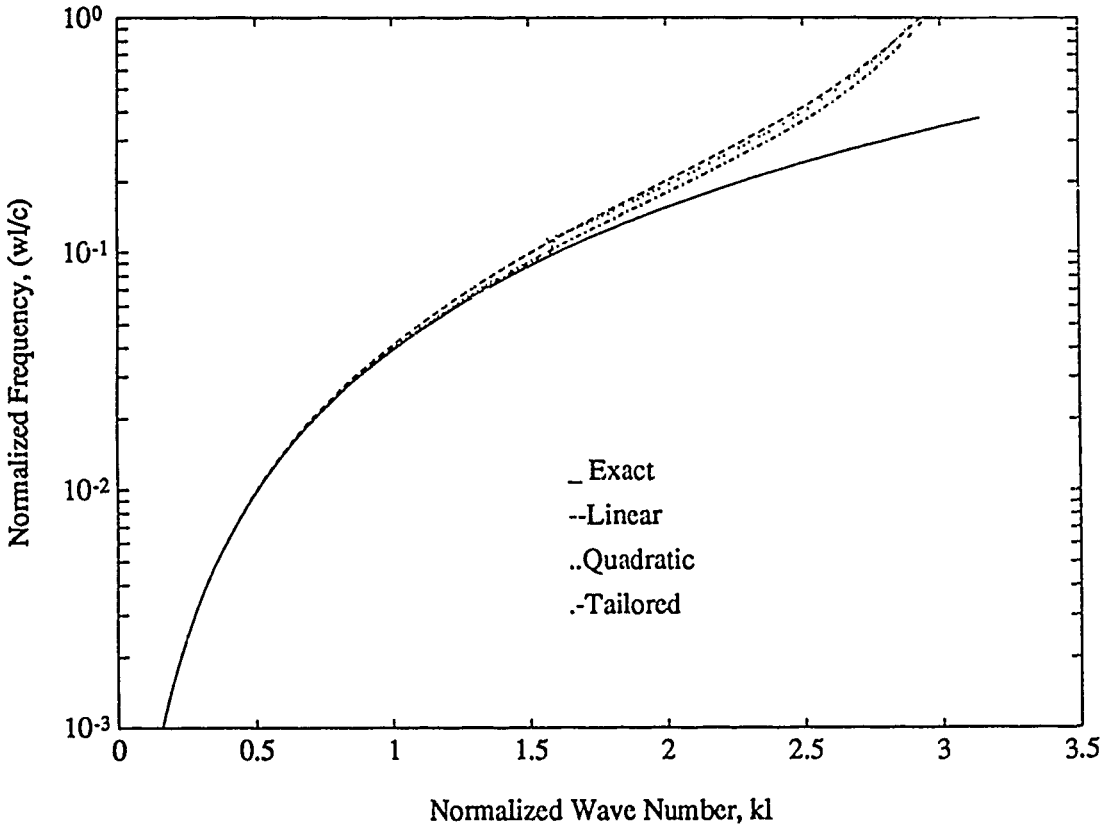


Figure 11: Fourier Analysis, Tailored Beam Element, Case I
Tailored Wave Range: $\frac{\pi}{6} \leq kl \leq \frac{\pi}{3}$
Parameters: $\alpha_m = 0.6692$, $\alpha_s = 1.0018$, $\alpha_b = 0.6244$

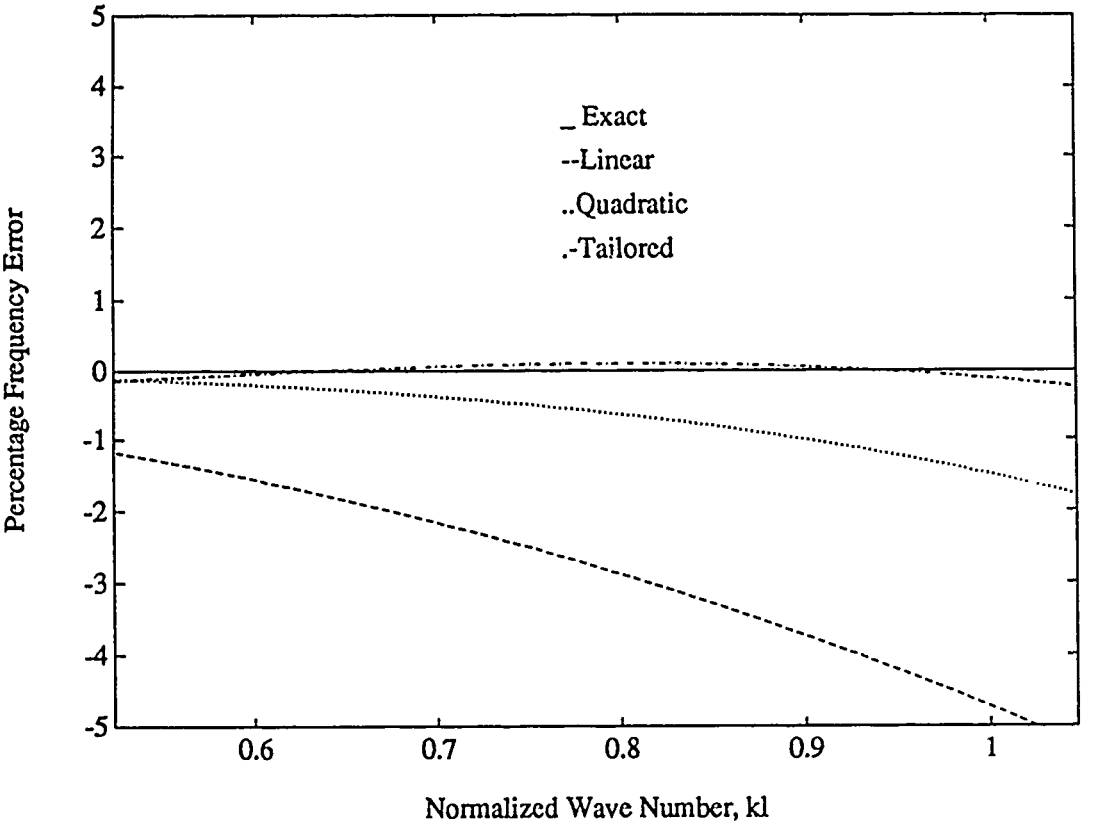


Figure 12: Frequency Error, Tailored Beam Element, Case I
Tailored Wave Range: $\frac{\pi}{6} \leq kl \leq \frac{\pi}{3}$
Parameters: $\alpha_m = 0.6692$, $\alpha_s = 1.0018$, $\alpha_b = 0.6244$

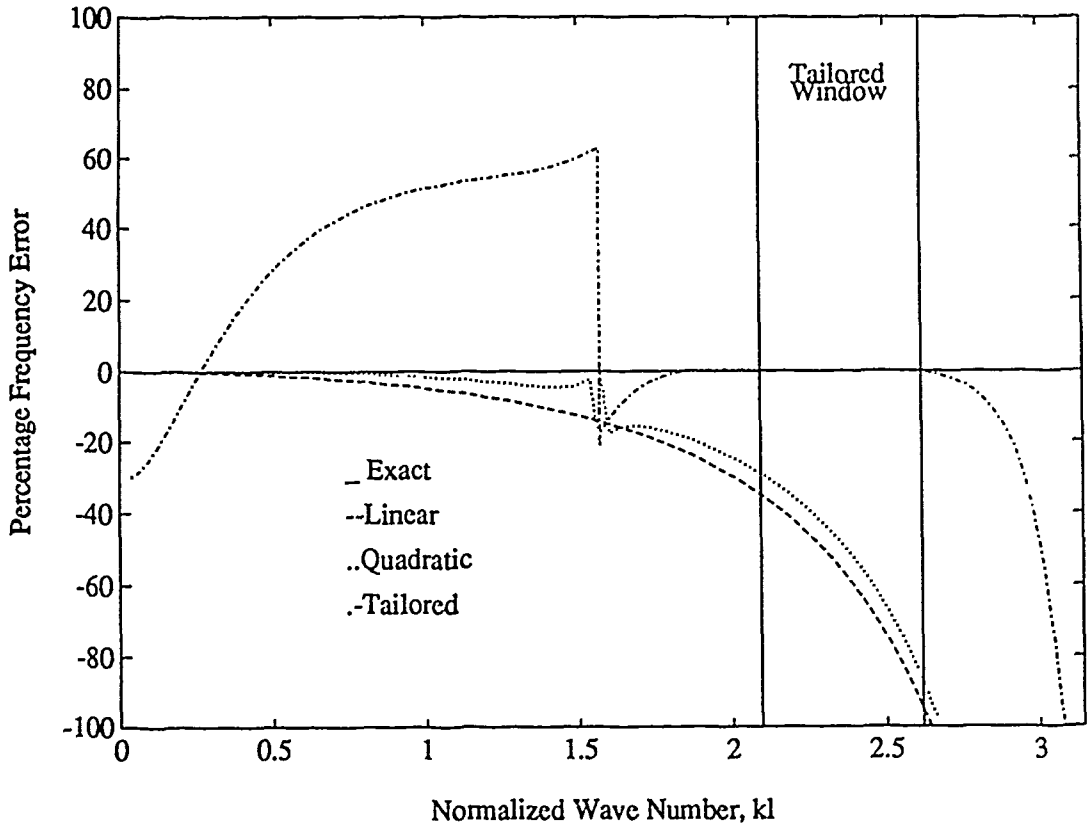


Figure 13: Frequency Error, Tailored Beam Element, Case II
Tailored Wave Range: $\frac{2\pi}{3} \leq kl \leq \frac{5\pi}{6}$
Parameters: $\alpha_m = 1.5321$, $\alpha_s = 1.0854$, $\alpha_b = -0.2374$

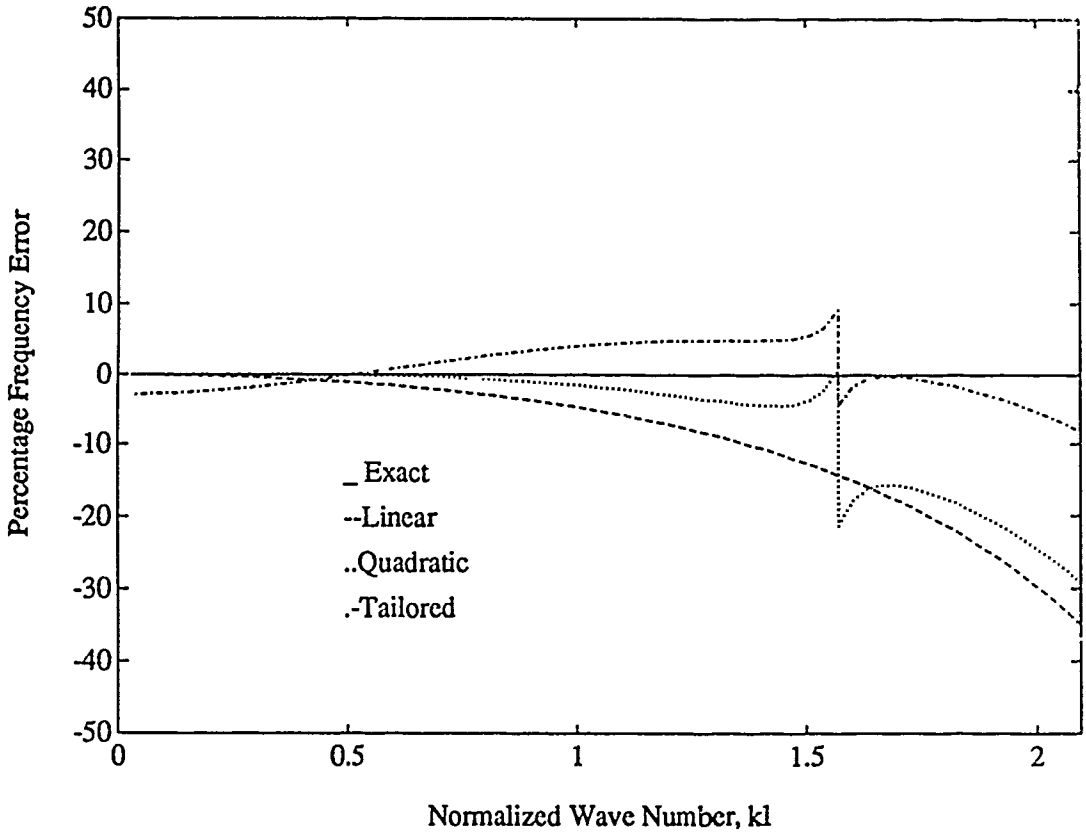


Figure 14: Frequency Error, Tailored Beam Element, Case III
Tailored Wave Range: $0 \leq kl \leq \frac{2\pi}{3}$
Parameters: $\alpha_m = 0.8508$, $\alpha_s = 1.0331$, $\alpha_b = 0.6685$

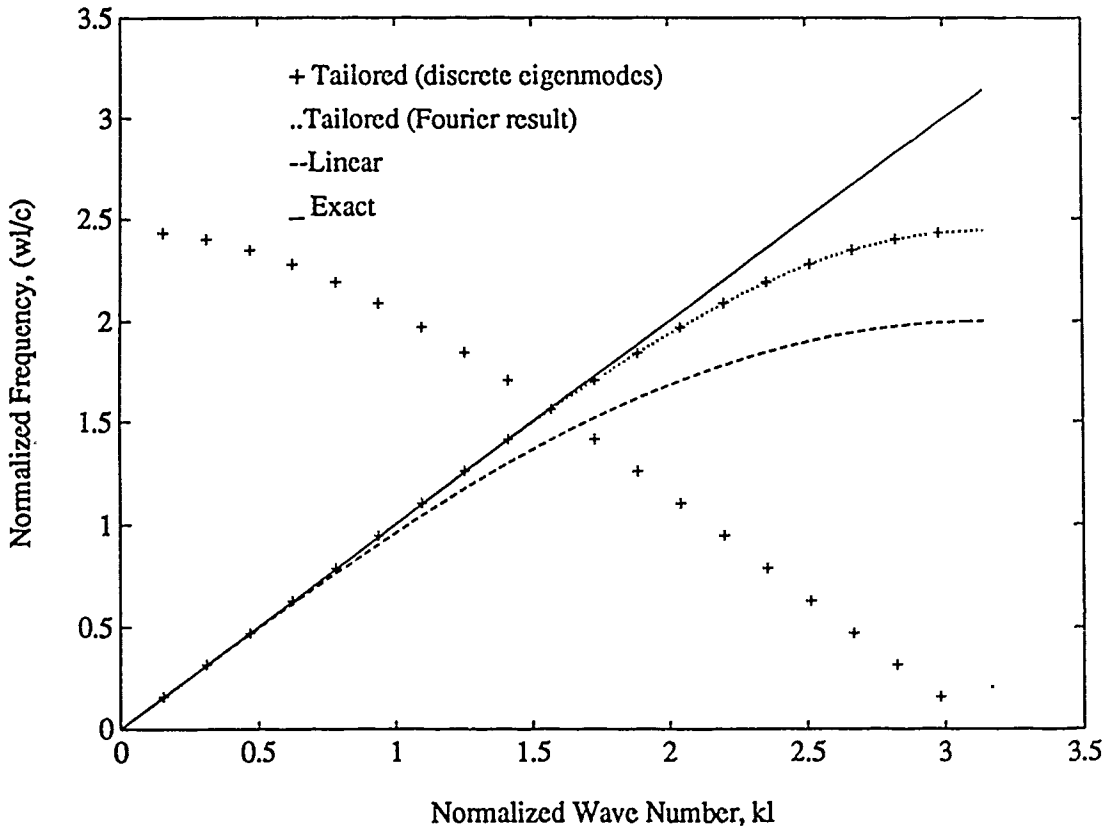


Figure 15: Comparison of Fourier Analysis to Discrete Bar Model, Case III
Tailored Wave Range: $\frac{\pi}{3} \leq kl \leq \frac{\pi}{2}$
Parameters: $\alpha_m = 0.5409$, $\alpha_k = 1.3188$

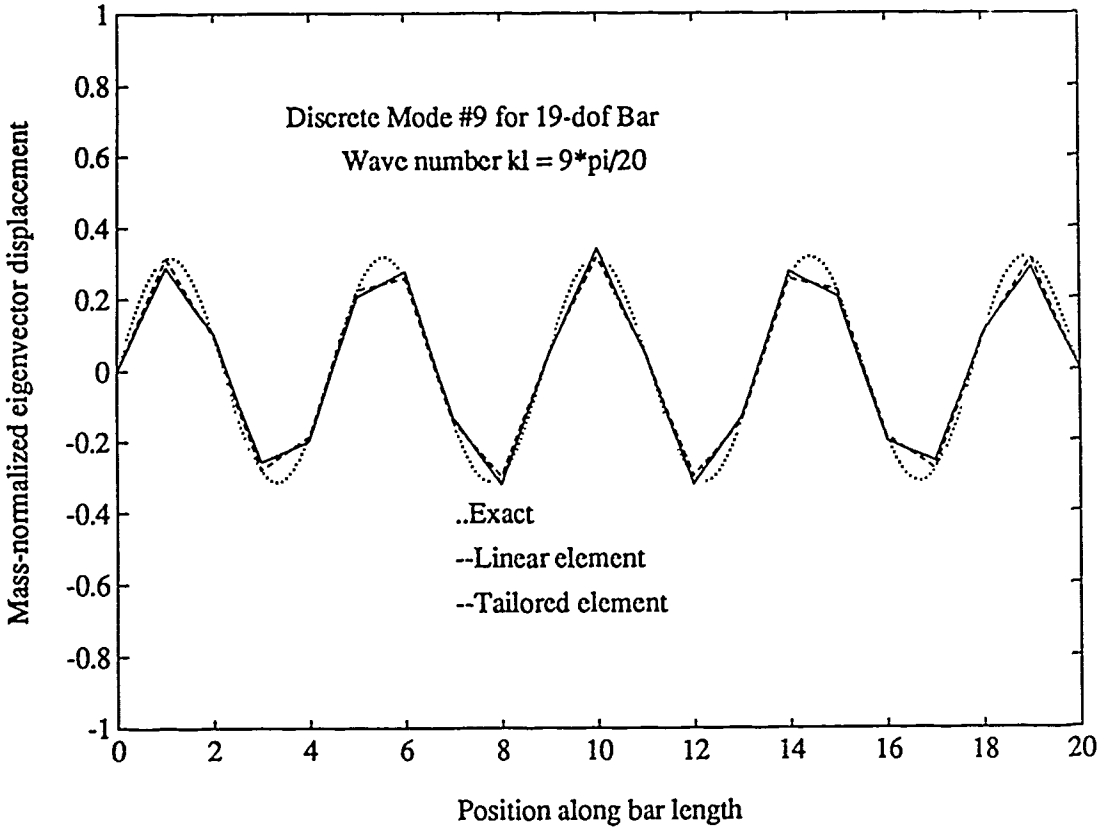


Figure 16: Low Frequecy Mode Shape for Discrete Bar Models, Case III
Tailored Wave Range: $\frac{\pi}{3} \leq kl \leq \frac{\pi}{2}$
Parameters: $\alpha_m = 0.5409$, $\alpha_k = 1.3188$

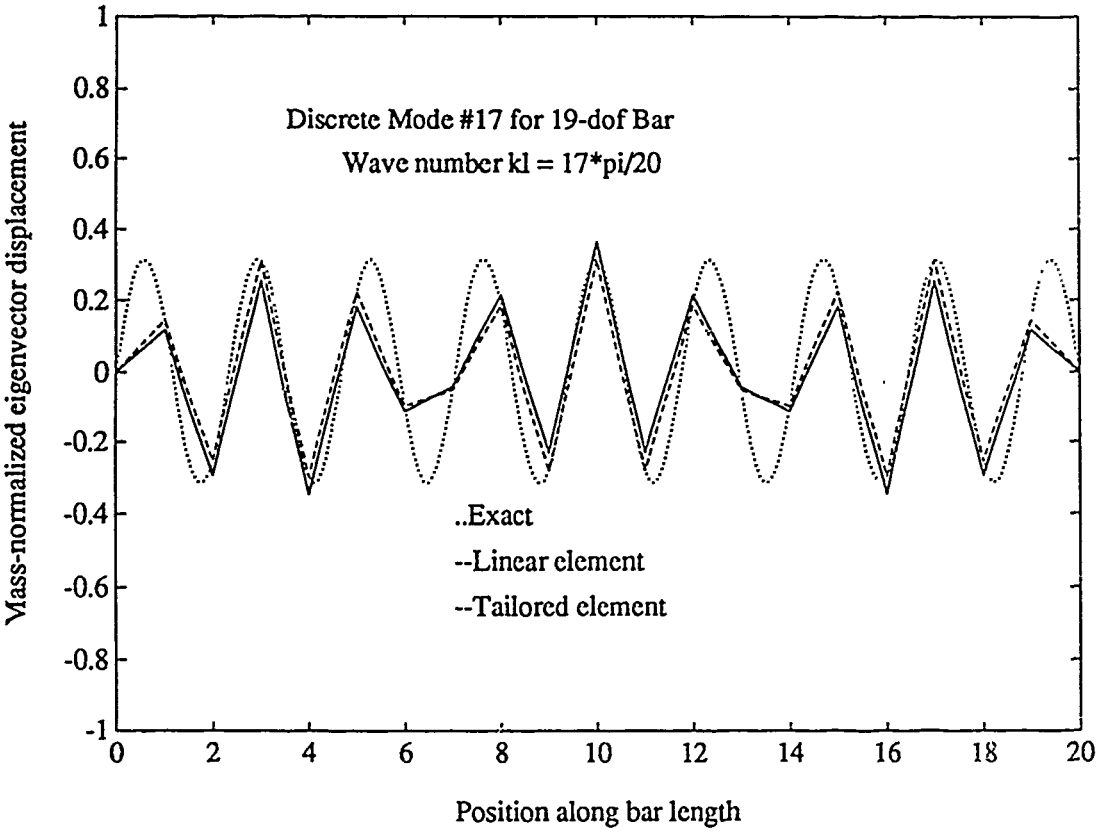


Figure 17: High Frequecy Mode Shape for Discrete Bar Models, Case III
Tailored Wave Range: $\frac{\pi}{3} \leq kl \leq \frac{\pi}{2}$
Parameters: $\alpha_m = 0.5409$, $\alpha_k = 1.3188$

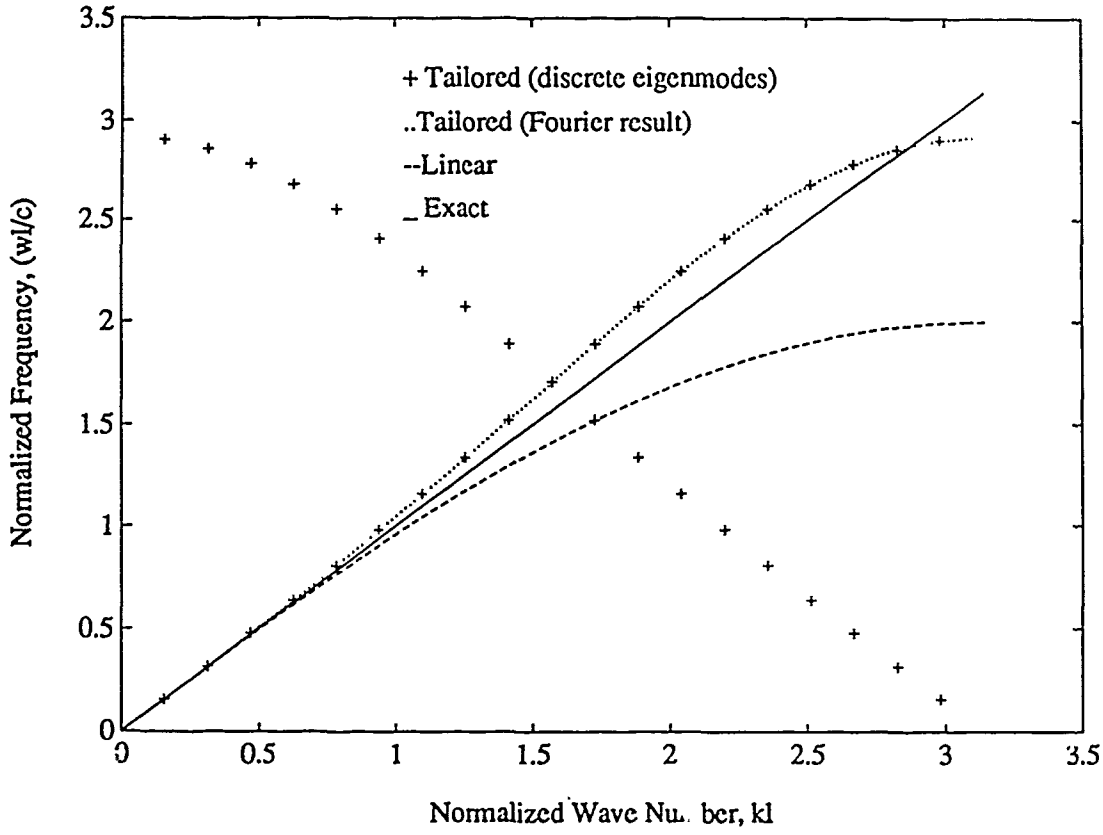


Figure 18: Comparison of Fourier Analysis to Discrete Bar Model, Case VI
Tailored Wave Range: $\frac{5\pi}{6} \leq kl \leq \pi$
Parameters: $\alpha_m = 0.9435$, $\alpha_k = 2.7517$

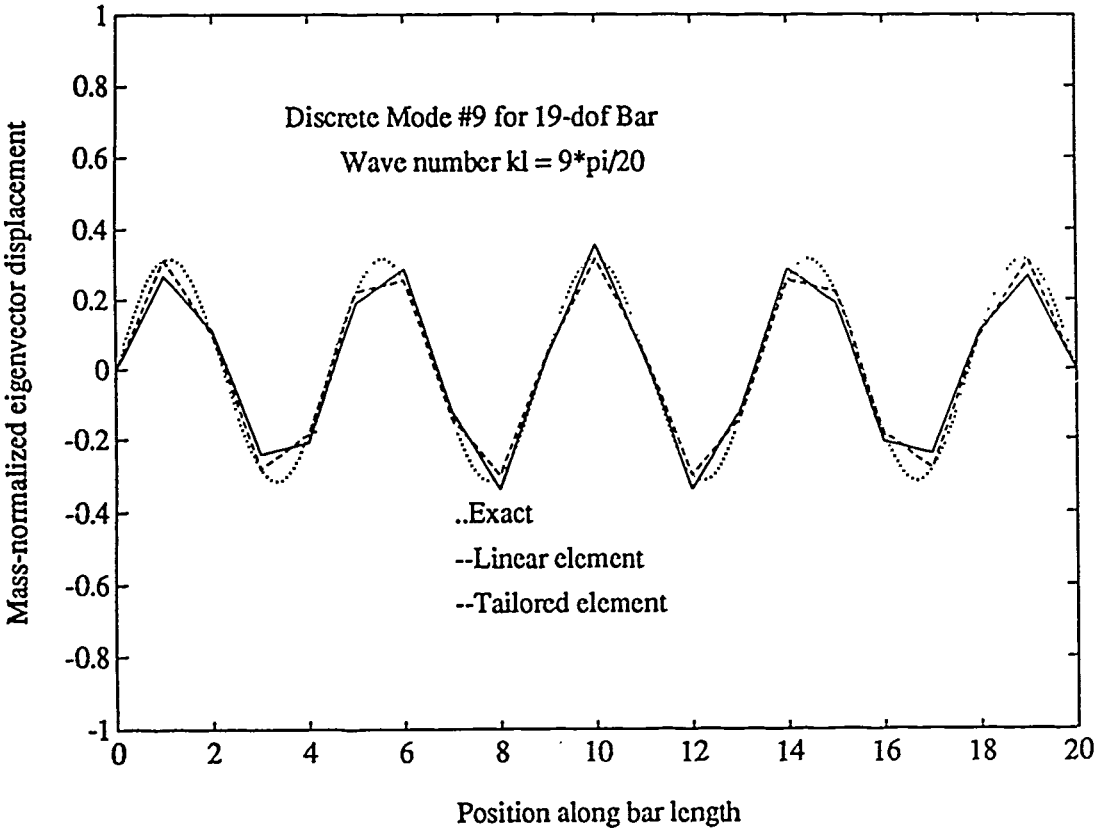


Figure 19: Low Frequecy Mode Shape for Discrete Bar Models, Case VI
Tailored Wave Range: $\frac{5\pi}{6} \leq kl \leq \pi$
Parameters: $\alpha_m = 0.9435$, $\alpha_k = 2.7517$

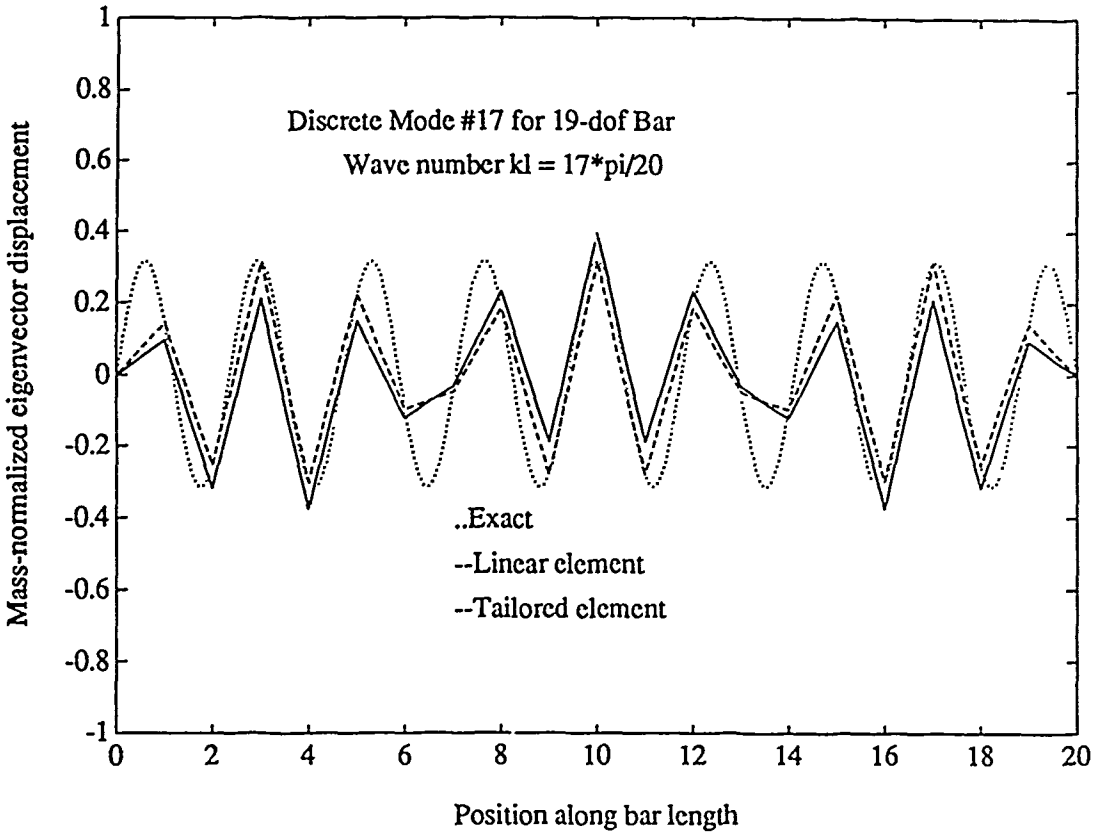


Figure 20: High Frequecy Mode Shape for Discrete Bar Models, Case VI
Tailored Wave Range: $\frac{5\pi}{6} \leq kl \leq \pi$
Parameters: $\alpha_m = 0.9435$, $\alpha_k = 2.7517$

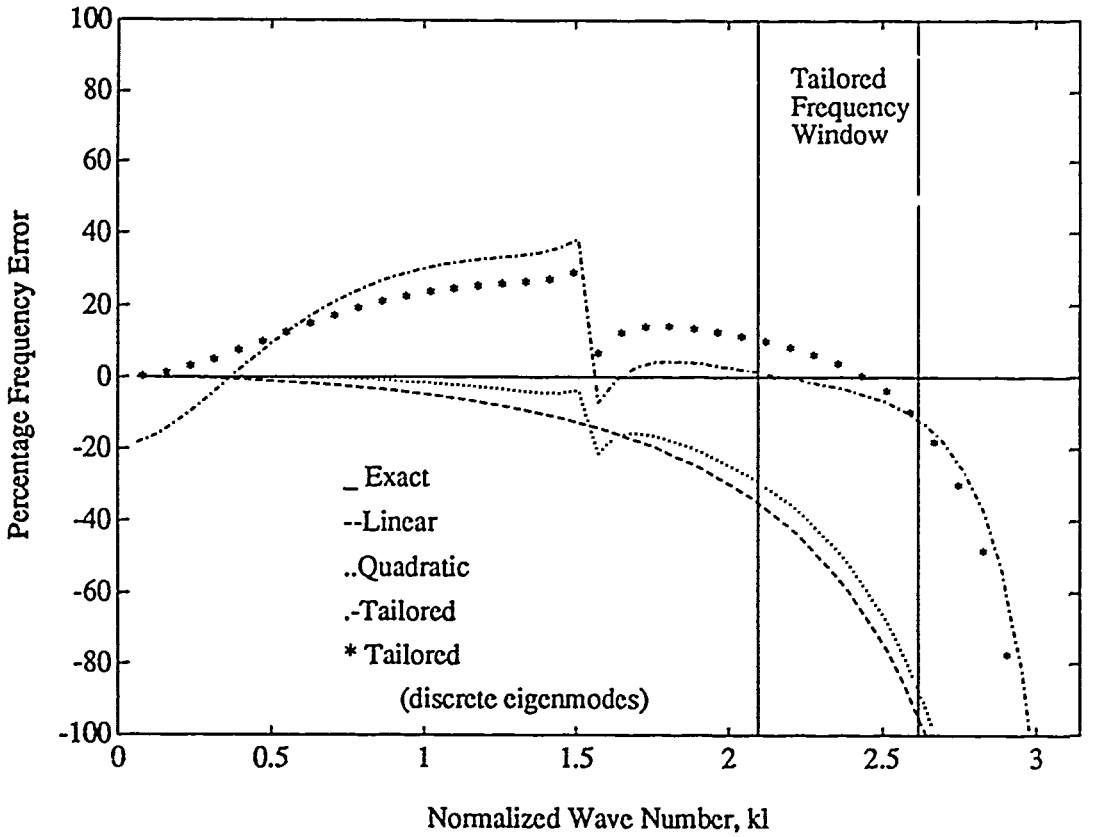


Figure 21: Comparison of Fourier Analysis to Discrete Beam Model, Case IIa
Tailored Wave Range: $\frac{2\pi}{3} \leq kl \leq \frac{5\pi}{6}$
Parameters: $\alpha_m = 1.5321$, $\alpha_s = 1.0800$, $\alpha_b = 0.5000$

A Systematic Determination of Lumped and Improved Consistent Mass Matrices for Vibration Analysis

89-1335-CP

K. C. Park[1] *and Daniel D. Jensen*[2]
Center for Space Structures and Controls
University of Colorado, Campus Box 429
Boulder, CO 80309
Phone: 303-492-6330

Abstract

A systematic procedure for determining the lumped mass matrix and improved consistent mass matrices has been proposed for vibration analyses by the finite element method. The procedure is based on the discrete Fourier analysis which enables one to compare the numerical approximations with the corresponding continuum characteristics. The procedure is applied to vibrations of bar, Euler-Bernoulli beam and plate bending elements. The results obtained by the present procedure clearly indicate that a judicious use of the improved mass matrices offered in the paper can lead to a significant accuracy improvement for intermediate frequencies that can play important roles in modeling of control-structure interaction systems, dynamic localizations and acoustic responses for space structures and underwater vehicles.

1. Introduction

The question of mass lumping or rather the systematic construction of mass matrix for the vibration and transient analysis of structures by the finite element method remains to date an unresolved issue. Apparently, it was Archer (1963) who first introduced a procedure for generating mass matrices based on the same displacement shape functions that are used in the construction of element stiffness matrices. The mass matrices generated according to Archer's procedure have become known as "consistent" mass matrices. In contrast to the consistent mass matrix, a diagonal or diagonalized mass matrix is refered to as a lumped mass matrix.

Even though the use of consistent mass matrices yields for most applications better accuracy in the frequency analysis, the lumped mass matrix continues to be prefered by the practicing engineers due to its computational simplicity and a data storage saving in the computer. Such attractive features of the lumped mass matrix motivated several investigators in the past to propose various mass lumping procedures such as a row sum of the consistent mass matrix (Leckie and Lindberg, 1963; Tong, Pian and Buciarelli, 1971; Krieg and Key, 1973), the use of a scaled diagonal entries from the stiffness matrix (Hinton et al, 1976), a selective sum of a low order-based consistent mass matrix (Fried and Malkus, 1975), and combinations of these.

Although existing mass lumping procedures are intuitively appealing for low-order elements such as constant strain bar, and beam bending elements, such intuitive (or ad-hoc) procedures become quickly ambiguous for high-order elements. As a result, at present no agreed-upon lumped mass matrices exist for cubic Euler-Bernoulli beams and for eight-noded serendipity plate/shell elements. Hence, there exists a lack of a systematic procedure for mass lumping.

The objective of the present paper is first to develop a systematic lumping procedure based on the discrete Fourier analysis of the finite element method (Park and Flaggs, 1984; Flaggs, 1988) and second to symbolically synthesize a series of more accurate mass matrices for vibration analysis when intermediate frequencies become important. To this end, the paper is organized as follows.

Section 2 revisits a discrete Fourier analysis of a bar modeled by the linear displacement approximation. A definition of the lumped mass matrix is proposed by comparing the characteristic equation for the continuum bar with that for the constant-strain bar. A simple synthesis of an improved mass matrix for the bar is proposed and its accuracy is assessed in terms of its wave dispersion curve. An example vibration problem with simply-fixed bar ends is analyzed, which demonstrates the systematic nature of the proposed definition of a lumped mass matrix and the general nature of discrete-Fourier synthesized mass matrices.

Section 3 applies the present definition of lumped mass matrices to a cubic Euler-Bernoulli beam element. A principal theory from the analysis of the cubic Euler-Bernoulli beam confirms the numerically well-known result that the lumping of the translational degree of freedom (w) and the neglect of the rotational freedom ($\frac{\partial w}{\partial x}$) yields a most accurate frequency prediction. The present theory succinctly illustrates that such a mass matrix is the only theoretically consistent approximation. A problem analyzed by Archer

[1] – Professor of Aerospace Engineering, University of Colorado. Member AIAA.

[2] – Graduate Research Assistant, Department of Aerospace Engineering Science.

Copyright © American Institute of Aeronautics and Astronautics, Inc., 1989. All rights reserved.

is revisited in order to asses our improved mass matrix. It is shown that the proposed synthesized mass matrix considerably improves the third and fourth frequencies for a two-element beam, thus establishing the soundness of the proposed synthesized mass matrix.

The present improved mass modeling for plate vibrations is presened in Section 4. To this end, a Fourier analysis of the frequency vs. wave number characteristics is carried out for an infinite plate of both the continuum and finite element approximation by a four-noded element. Such a Fourier analysis is believed to provide insight into the accuracy of vibration analysis for an infinite plate. In oder to utilize the mass matrix modeling based the discrete Fourier analysis, finite element plate vibrations with free edges have been performed with increasing meshes. The results obtained from the discrete Fourier analysis and numerical tests for free-edge plate vibrations indicate that a best accuracy for an infinite plate, when analyzed by a four-node element, is achieved by an average of the lumped and the consistent matrices. For plate with finite dimensions, a best accuracy is achievable with quarter of the lumped mass and three quarter of the consistent mass matrix.

2. A Proposition for Lumped Mass Matrix

A mass-matrix lumping procedure that we are about to propose is based on the discrete Fourier analysis. Since an easy example of Fourier analysis that one can perform is the continuum equation for a bar and its discrete counterpart, we will first introduce their Fourier analyses. We will then identify the Fourier operators for mass matrices. In a simplest term, our proposed lumped mass matrix is defined as follows:

> *Let k be the wave number and $\bar{M}_{con}$ the Fourier-transformed consistent mass matrix, then the lumped mass matrix, $\bar{M}_{lump}$, in the Fourier domain is defined by*
>
> $$\bar{M}_{lump} = \lim_{k \to 0} \bar{M}_{con} \qquad (2 \cdot 1)$$

We will now illustrate our proposition for mass-lumping procedure via a bar element.

The equation of motion for a uniform elastic bar can be written as

$$\rho \frac{\partial^2 u}{\partial t^2} = E \frac{\partial^2 u}{\partial^2 x} \qquad (2 \cdot 2)$$

where ρ, E, u, x are the density, Young's modulus, the displacement and the coordinate, respectively.

The traditional Fourier analysis begins by seeking a general harmonic wave solution of Eq. (2.2) of the form

$$u = \bar{u} e^{i(\omega t - kx)} \qquad (2 \cdot 3)$$

with ω being the circular frequency, k, the corresponding wave number and $i = \sqrt{-1}$. Substitution of Eq. (2.3) into Eq. (2.2) yields

$$L(\omega, k) \cdot \bar{u} = 0 \qquad (2 \cdot 4)$$

where the Fourier operator, $L(\omega, k)$, is given explicity by

$$L(\omega, k) = -\rho\omega^2 + Ek^2 \qquad (2 \cdot 5)$$

For our subsequent discussions, we reexpress the above equation as:

$$\begin{cases} L(\omega, k) = -\omega^2 \bar{M}_{exact} + \bar{K}, \\ \text{with} \quad \bar{M}_{exact} = \rho \cdot 1, \quad \bar{K} = Ek^2 \end{cases} \qquad (2 \cdot 6)$$

A corresponding discrete Fourier analysis can be performed when the bar equation (2.2) is approximated by the finite element method, which has been studied by many investigators (Bazant, 1978; Belytschko and Mullen, 1978; Vichnevetsky, 1982; Park and Flaggs, 1984; Celep and Turhan, 1987; Flaggs, 1988). A preoccupation of these studies, however, was to address the effect of internal energy discretizations on the wave propagation characteristics.

In the present study, we will examine the effect of kinetic energy discretizations on the frequency characteristics and deduce from such a study the proposed mass-lumping procedure as well as a synthesis procedure to obtain improved consistent mass matrices. To this end, let us revisit a constant bar element. An elementary finite element implementation gives for a bar element of a uniform length, ℓ, the following discrete equation for an interior node, m (e.g., Park and Flaggs, 1984):

$$\frac{\rho}{6}(\ddot{u}_{m-1} + 4\ddot{u}_m + \ddot{u}_{m+1}) + \frac{E}{l^2}(u_{m-1} - 2u_m + u_{m+1}) = 0 \qquad (2 \cdot 7)$$

Substitution of Eq. (2.3) into Eq. (2.7) yields the discrete Fourier operator

$$L^D_{con}(\omega, k) = -\omega^2 \bar{M}_{con} + \bar{K}^D \qquad (2 \cdot 8)$$

where

$$\bar{M}_{con} = \rho \cdot (1 - \frac{\ell^2}{6}\bar{k}^2), \quad \bar{K}^D = E\bar{k}^2 \qquad (2 \cdot 9)$$

in which the discrete wave number, $\bar{k}$, is defined as

$$\bar{k} = \frac{\sin \frac{kl}{2}}{\frac{l}{2}}. \qquad (2 \cdot 10)$$

For a lumped mass matrix, we have the following discrete Fourier operator:

$$L^D_{lump}(\omega, k) = -\omega^2 \bar{M}_{lump} + \bar{K}^D \qquad (2 \cdot 11)$$

where

$$\bar{M}_{lump} = \rho \cdot 1 \qquad (2 \cdot 12)$$

Test of Proposed Mass-Lumping Procedure (2.1):

Note that $\bar{M}_{con}$ can be expanded to read

$$\bar{M}_{con} = \rho \cdot (1 - \frac{4}{6}\sin^2 \frac{kl}{2}) \qquad (2 \cdot 13)$$

Hence, we have

$$\lim_{k\to 0} \tilde{\mathbf{M}}_{con} = \rho \cdot 1 = \tilde{\mathbf{M}}_{lump} \qquad (2\cdot 14)$$

which proves our proposition (2.1) to be valid (at least for the bar!).

We now address our second related task: a systematic way of constructing consistent mass matrices that can lead to improved vibration analysis. To this end, we note that the characteristic equation that relates the wave number (k) to the frequency (ω) is obtained by setting $L(\omega, k) = 0$ for the continuum case:

$$(\frac{\omega}{c})^2 = k^2 \qquad (2\cdot 15)$$

where the wave speed, c, defined as $c = \sqrt{E/\rho}$ is constant.

The characteristic equation, Eq. (2.15), indicates that for the continuum solution, the wave number, k, is directly proportional to the frequency, ω, i.e., $k = \omega/c$. With c constant, each Fourier component of a wave group will propagate without dispersion with the same phase velocity.

To examine the effect of kinetic energy discretizations on the accuracy of vibration analysis, we propose the following simple modification for our proposed mass matrix:

$$\tilde{\mathbf{M}}_{new} = (1-\alpha)\cdot\tilde{\mathbf{M}}_{lump} + \alpha\cdot\tilde{\mathbf{M}}_{con} \qquad (2\cdot 16)$$

where α is a constant to be determined. Note that $\alpha = 0$ corresponds to the case of lumped mass matrix and $\alpha = 1$ to consistent mass matrix, respectively. Figure 1 illustrates frequency vs wave number for the continuum bar, the discrete bar with the consistent mass matrix ($\alpha = 1$) and with the lumped mass ($\alpha = 0$), and for an averaged mass matrix($\alpha = 0.5$). Although not shown in the figure, other values of α can be selected in order to tailor the accuracy requirement for different frequency ranges. For example, $\alpha = (0.5, 0.56829159, 0.89207289)$ gives a most accurate result for small k, k around $\pi/2$ and k around π, respectively. Hence, depending upon the critical accuracy range of interest, one can adjust the mass matrix accordingly.

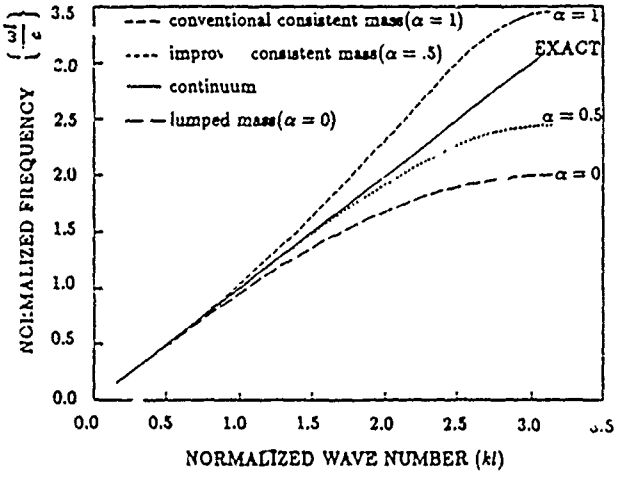


Figure 1. Dispersion Curve for Linear Bar

3. Mass-Lumping of a Euler-Bernoulli Beam Element

In the preceding section we have succintly demonstrated that a discrete Fourier analysis can provide numerical and physical insight into mass lumping as well as can lead to improved mass matrix approximation. In this section, we will demonstrate that the proposed mass lumping procedure is also applicable for cases that require discrete Fourier matrix operators.

The homogeneous differential equations of motion for the Euler-Bernoulli beam can be expressed as

$$\rho A\frac{\partial^2 w}{\partial t^2} + EI\frac{\partial^4 w}{\partial x^4} = 0 \qquad (3\cdot 1)$$

where A is the cross-section area of the beam, I is the moment of inertia. With a general harmonic wave solution of (3.1) of the form

$$w = \tilde{w}e^{i(\omega t - kx)} \qquad (3\cdot 2)$$

we obtain the Fourier operator for the beam as

$$L(\omega, k) = -\omega^2\cdot\mathbf{M}_{exact} + \tilde{K}_{beam} \qquad (3\cdot 3)$$

where

$$\mathbf{M}_{exact} = \rho A, \quad \tilde{K}_{beam} = EIk^4 \qquad (3\cdot 4)$$

A cubic interpolation of w gives for each beam element of length ℓ the following consistent mass and stiffness matrices

$$\mathbf{M}^e_{const} = \frac{\rho A\ell}{420}\begin{bmatrix} 156 & 22\ell & 54 & -13\ell \\ 22\ell & 4\ell^2 & 13\ell & -3\ell^2 \\ 54 & 13\ell & 156 & -22\ell \\ -13\ell & -3\ell^2 & -22\ell & 4\ell^2 \end{bmatrix} \qquad (3\cdot 5)$$

$$\mathbf{K}^e_{beam} = \frac{EI}{\ell^3}\begin{bmatrix} 12 & 6\ell & -12 & 6\ell \\ 6\ell & 4\ell^2 & -6\ell & 2\ell^2 \\ -12 & -6\ell & 12 & -6\ell \\ 6\ell & 2\ell^2 & -6\ell & 4\ell^2 \end{bmatrix} \qquad (3\cdot 6)$$

where the elemental nodal degrees of freedom are arranged as

$$\mathbf{u}^e = \lfloor w_1, \theta_1, w_2, \theta_2 \rfloor^T \qquad (3\cdot 7)$$

By assembling two interior beam elements and designating their nodes, $m-1, m$ and $m+1$, respectively, we obtain the following difference equations for the mth node:

$$\frac{\rho A\ell}{420}\{(54\dot{w}_{m-1} + 312\dot{w}_m + 54\dot{w}_{m+1}) + 13\ell(\ddot{\theta}_{m-1} - \ddot{\theta}_{m+1})\}$$
$$+\frac{EI}{\ell^3}\{12(-w_{m-1} + 2w_{m-1} - w_{m+1}) + 6\ell(-\theta_{m-1} + \theta_{m+1})\} = 0 \qquad (3\cdot 8)$$

$$\frac{\rho A\ell}{420}\{13\ell(-\dot{w}_{m-1} + \dot{w}_{m+1}) + \ell^2(-3\ddot{\theta}_{m-1} + 8\ddot{\theta}_m - 3\ddot{\theta}_{m+1})\}$$
$$+\frac{EI}{\ell^3}\{6\ell(w_{m-1} - w_{m+1}) + 2\ell^2(\theta_{m-1} + 4\theta_m + \theta_{m+1})\} = 0 \qquad (3\cdot 9)$$

Since w_j and θ_j are treated independently, we seek a solution

$$\begin{Bmatrix} w_j \\ \theta_j \end{Bmatrix} = \begin{Bmatrix} \tilde{w}_j \\ \tilde{\theta}_j \end{Bmatrix} e^{i(\omega t - kx)} \qquad (3\cdot 10)$$

to Fourier-transform the coupled difference equation (3.8) and (3.9) to obtain

$$(-\omega^2 \tilde{M}_{con} + \tilde{K}_{beam}) \begin{Bmatrix} \tilde{w}_m \\ \tilde{\theta}_m \end{Bmatrix} = 0 \qquad (3\cdot 11)$$

where

$$\tilde{M}_{con} = \rho A \ell \begin{bmatrix} \tilde{m}_{11} & i\tilde{m}_{12} \\ -i\tilde{m}_{12} & \ell^2 \tilde{m}_{22} \end{bmatrix} \qquad (3\cdot 12)$$

$$\tilde{K}_{beam} = \frac{12EI}{\ell^3} \begin{bmatrix} \bar{k}^2\ell^2 & i\ell \sin k\ell \\ -i\ell \sin k\ell & \ell^2(1 - \frac{\ell^2}{6}\bar{k}^2) \end{bmatrix} \qquad (3\cdot 13)$$

in which

$$\begin{cases} \tilde{m}_{11} = (1 - \frac{9}{70}\bar{k}^2\ell^2), & \tilde{m}_{12} = -\frac{13}{210}\ell \sin k\ell, \\ \tilde{m}_{22} = \frac{1}{210}(1 + 1.5\bar{k}^2\ell^2) \end{cases} \qquad (3\cdot 14)$$

The lumped-mass matrix, according to our proposition (2.1), for the mth assembled node thus becomes

$$\tilde{M}_{lump} = \lim_{k\to 0} \tilde{M}_{con} = \rho A \ell \begin{bmatrix} 1 & 0 \\ 0 & \frac{\ell^2}{210} \end{bmatrix} \qquad (3\cdot 15)$$

which, when translated into the element mass matrix, is equivalent to

$$\mathbf{M}^e_{lump} = \rho A \ell \begin{bmatrix} \frac{1}{2} & & & \\ & \frac{13}{420} & & \\ & & \frac{1}{2} & \\ & & & \frac{\ell^2}{420} \end{bmatrix} \qquad (3\cdot 16)$$

Remark: One of ad-hoc mass-lumping procedures is to sum up d.o.f.-by-d.o.f. contribution. When this ad-hoc technique is applied to the element mass matrix (3.5), one obtains:

$$\mathbf{M}^e_{lump} = \mathbf{M}^w_{lump} + \mathbf{M}^\theta_{lump} \qquad (3\cdot 17)$$

where for the w-d.o.f.s we have

$$\mathbf{M}^w_{lump} = \frac{\rho A\ell}{420} \begin{bmatrix} 156 & 0 & 54 & 0 \\ 0 & 0 & 0 & 0 \\ 54 & 0 & 156 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \simeq \rho A\ell \begin{bmatrix} \frac{1}{2} & 0 & & \\ & & \frac{1}{2} & \\ & & & 0 \end{bmatrix} \qquad (3\cdot 18)$$

For the θ-d.o.f.s

$$\mathbf{M}^\theta_{lump} = \frac{\rho A\ell}{420} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 4\ell^2 & 0 & -3\ell^2 \\ 0 & 0 & 0 & 0 \\ 0 & -3\ell^2 & 0 & 4\ell^2 \end{bmatrix} \simeq \frac{\rho A\ell}{420} \begin{bmatrix} 0 & & & \\ & \ell^2 & & \\ & & 0 & \\ & & & \ell^2 \end{bmatrix} \qquad (3\cdot 19)$$

Hence, for simple elements the ad-hoc d.o.f.-by-d.o.f. row-summation procedure is justified in view of the present proposed mass-lumping procedure (2.1). It should be mentioned that care must be exercised in lumping matrices for higher-order elements.

Now, to address the accuracy associated with the choice of a mass matrix (either $\tilde{M}_{lump}$ or $\tilde{M}_{con}$, or even their combinations), Let us examine the characteristic equation from (3.11), viz,

$$\| \begin{bmatrix} -\omega^2\tilde{m}_{11} + a^2\bar{k}^2\ell^2 & -i\omega^2\tilde{m}_{12} + ia^2\ell \sin k\ell \\ i\omega^2\tilde{m}_{12} - ia^2\ell \sin k\ell & -\omega^2\ell^2\tilde{m}_{22} + a^2\ell^2(1 - \frac{\ell^2}{6}\bar{k}^2) \end{bmatrix} \| = 0,$$

$$a^2 = \frac{12EI}{\rho A\ell^4} \qquad (3\cdot 20)$$

which yields the following frequency vs. wave number equation:

$$\begin{cases} (\tilde{m}_{11}\tilde{m}_{22} - \tilde{m}^2_{12})\omega^4 - a^2(\tilde{m}_{11}\tilde{I}_\theta + 2\sin^2 k\ell \cdot \tilde{m}_{12} + \tilde{m}_{22}\bar{k}^2\ell^2)\omega^2 \\ +a^4\frac{1}{12\bar{k}^4\ell^4} = 0 \end{cases} \qquad (3\cdot 21)$$

where $\tilde{I}_\theta = 1 - \frac{\ell^2}{6}\bar{k}^2$.

Comparing (3.21) with its differential equation counterpart (3.3), one concludes that, since $\tilde{m}_{11}$ is the translational part, we must have

$$\tilde{m}_{12} = 0 \quad \text{and} \quad \tilde{m}_{22} = 0 \qquad (3\cdot 22)$$

if (3.21) is to have the following form:

$$-\omega^2 \mathbf{M}_{lump} + \tilde{K}_{beam} \simeq 0 \qquad (3\cdot 23)$$

Hence, we conclude that for the cubic Euler-Bernoulli beam element to be consistent with the original differential equation, one must employ the following mass matrix (i.e., $\tilde{m}_{12} = 0$, $\tilde{m}_{22} = 0$):

$$\mathbf{M}^e = \frac{\rho A\ell}{420} \begin{bmatrix} 210 - 54\alpha & 0 & 54\alpha & 0 \\ 0 & 0 & 0 & 0 \\ 54\alpha & 0 & 210 - 54\alpha & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad 0 \le \alpha \le 1 \qquad (3\cdot 24)$$

Application of (3.24) yields the following characteristic equation:

$$-\omega^2(1 - \frac{\alpha}{6}\ell^2\bar{k}^2) + \frac{EI}{\rho A\ell^4}\bar{k}^4\ell^4 = 0 \qquad (3\cdot 25)$$

so that we have from (3.3) and (3.25) the following frequency equations

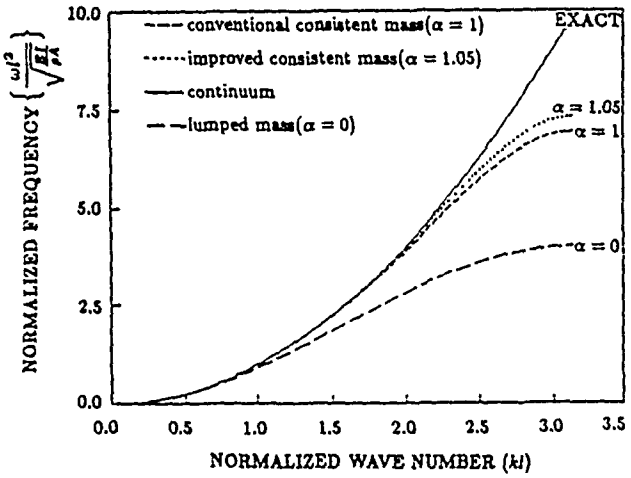


Figure 2. Dispersion Curve for Beam with Rotatory Inertia

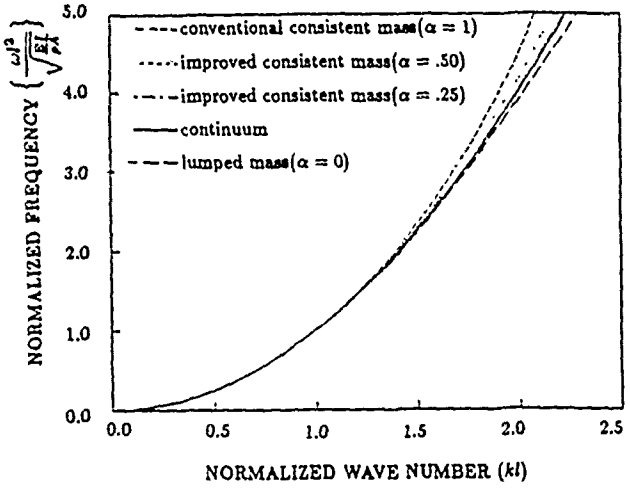


Figure 3. Dispersion Curve for Beam without Rotatory Inertia

For continuum case:

$$\frac{\omega}{\sqrt{\frac{EI}{\rho A \ell^4}}} = (\bar{k}\ell)^2 \qquad (3 \cdot 26)$$

For the consistent matrix (3.23):

$$\frac{\omega}{\sqrt{\frac{EI}{\rho A \ell^4}}} = \frac{(\bar{k}\ell)^2}{(1 - \frac{\alpha}{6}\ell^2\bar{k}^2)^{\frac{1}{2}}} \qquad (3 \cdot 27)$$

Figure 2 shows the frequency vs. the normalized wave number for the mass matrix (3.24), that is, neglecting the rotational contribution to the element mass matrix (see Eq. (3.25)). For this case, the larger the value of α, the more accurate the frequency curve becomes.

Of course, as shown by Archer (1963), the use of the consistent mass matrix (3.5) may improve the frequency accuracy over the lumped mass matrix (3.16). In order to gain insight into the role of various mass modeling on the accuracy of vibration frequencies and mode shapes, we apply our proposed mass matrix formula (2.16) by combining (3.5) and (3.16) to obtain:

$$M^* = \frac{\rho A \ell}{420}\begin{bmatrix} 210-54\alpha & 22\ell\alpha & 54\alpha & -13\ell\alpha \\ 22\ell\alpha & \ell^2(1+3\alpha) & 13\ell\alpha & -3\ell^2\alpha \\ 54\alpha & 13\ell\alpha & 210-54\alpha & -22\ell\alpha \\ -13\ell\alpha & -3\ell^2\alpha & -22\ell\alpha & \ell^2(1+3\alpha) \end{bmatrix} \qquad (3 \cdot 28)$$

Figure 3 illustrate the effect of the mass-matrix averaging based on the above averaged mass matrix (3.28). Application of the above mass matrix yields a characterstic equation that is similar to (3.21) except $\bar{m}_{ij}$ are modified accordingly. As was the case for the bar, $\alpha = (0, 1)$ corresponds to the lumped matrix(3.16) and the consistent matrix(3.5), respectively. It is observed that for $\alpha = 0.25$ the discrete frequency vs the wave number curve follows almost on top of the continuum case.

In order to assess the preceding *a priori* determination of improved mass matrix based soley on the symbolic analysis, we have performed the vibration analysis of a free-free beam modeled by two elements and compared the present results with the one performed by Archer(1963). The results are summarized in Table 1. It is observed that the conventional consistent mass matrix ($\alpha = 1$) gives an error of about 1% for the first mode, 13% for the second mode, 45% for the third mode and 40% for the fourth mode. In contrast, for an averaged mass matrix ($\alpha = 0.5$), the corresponding errors are 27% for the first mode, 2.5% for the second mode, 1.8% for the third mode and 1.7% for the fourth mode, respectively. It is noted that the symbolic analysis results predict $\alpha = 0.5$ to be most accurate while the finite element solutions indicate that $\alpha = 0.25$ to be most accurate. We conjecture that this discrepancy is due to the fact that the finite element solutions are for finite beams whereas the Fourier analysis assumes an infinite beam. Neverthless, the accuracy prediction based on the discrete Fourier analysis as given in Figure 3 provides a qualitative measure of different mass modeling choices.

SOLUTION TYPE	ω_1	ω_2	ω_3	ω_4
EXACT	5.5944	15.481	30.266	49.945
$\alpha = 0.0$	3.4273	28.983	41.428	50.438
$\alpha = .25$	3.7075	19.101	32.911	50.204
$\alpha = .50$	4.0924	16.377	30.821	50.794
$\alpha = .75$	4.6610	15.867	32.474	53.599
$\alpha = 1.0$	5.6058	17.544	43.870	70.087

Table 1. Frequencies Computed by Different Mass Models for Beam

Incidently, the large error of the averaged mass matrix for the first mode is not too much of concern because as the number of elements are increased, the first mode quickly

converges. In fact, when five elements are assembled, the error for the first mode reduced to less than 0.2% with $\alpha = 0.5$ while maintaining high-accuracy for the higher modes. On the other hand, the conventional mass matrix ($\alpha = 1$) continues to give large errors for the higher modes.

4. Mass Matrices for Plate Vibration Analysis

So far we have shown that the proposed mass-lumping procedure (2.1) and the improved consistent mass matrix (2.16) lead to substantial improvements in beam vibration analyses. In particular, depending upon one's desire for tailoring the analysis accuracy for certain frequency ranges, one can modify *a priori* the mass matrix as proposed by (2.16) and manifested in Figs. 1 – 3. For Lagrange family of plate and shell elements, the proposed lumping procedure (2.1) is equally applicable. In this section we will first examine plate vibrations based on the Reissner-Mindlin plate theory. We will then employ a discrete counterpart of the continuum theory when the plate is approximated by a four-noded plate bending element. We will then compare the continuum and symbolically generated discrete equations in the Fourier domain in order to gain insight into the effect of mass matrices on the accuracy of vibration frequencies. We have found that our symbolic analysis of the discrete plate equations obtained from the finite element plate bending equations provides a qualitative measure of solution accuracy. Numerical experiments corroborate symbolic analysis results; for the case of the four-noded plate bending element, the mass matrix that yields a best accuracy is determined to be

$$\tilde{\mathbf{M}}_{new} = 0.25\tilde{\mathbf{M}}_{lump} + 0.75 \cdot \tilde{\mathbf{M}}_{con} \qquad (4 \cdot 1)$$

We now present a detailed analysis and some numerical results.

The Reissner-Mindlin plate equations can be expressed in differential matrix form as (Park and Flaggs, 1985):

$$\mathbf{L}(x,y,t) \cdot \mathbf{u} = \mathbf{f} \qquad (4 \cdot 2)$$

where

$$\mathbf{u}^T = \lfloor \theta, \phi, w \rfloor, \quad \mathbf{f}^T = \lfloor 0, 0, q \rfloor \qquad (4 \cdot 3)$$

$$\mathbf{L}(x,y,t) = \begin{bmatrix} D\frac{\partial^2}{\partial x^2} + D_{11}\frac{\partial^2}{\partial y^2} - D_s - I\frac{\partial^2}{\partial t^2} & D_{12}\frac{\partial^2}{\partial x \partial y} & D_s\frac{\partial}{\partial x} \\ & D\frac{\partial^2}{\partial y^2} + D_{11}\frac{\partial^2}{\partial x^2} - D_s - I\frac{\partial^2}{\partial t^2} & D_s\frac{\partial}{\partial y} \\ sym. & & -D_s\nabla^2 + m\frac{\partial^2}{\partial t^2} \end{bmatrix} \qquad (4 \cdot 4)$$

in which ∇^2 is defined as

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \qquad (4 \cdot 5)$$

and

$$D = \frac{Eh^3}{12(1-\nu^2)}, \quad D_{11} = \frac{1-\nu}{2}D,$$

$$D_{12} = \frac{1+\nu}{2}D, \quad D_s = \frac{\kappa E h}{2(1+\nu)} \qquad (4 \cdot 6)$$

in which θ, ϕ and w are the rotations and the transverse displacement; q is the transverse load per unit area; E, h, ν and κ are Young's modulus, plate thickness, Poisson's ratio and the shear correction factor; I and m are the rotatory inertia and the plate mass with x, y and t denoting the Cartesian coordinates and time, respectively.

The Fourier-transformed matrix operator, $\tilde{\mathbf{L}}(k_x, k_y, \omega)$, can be obtained for the plate equations by seeking a harmonic solution of the form

$$\mathbf{u}(x,y,t) = \mathbf{u}(x_0, y_0, t_0) \exp[j(\omega t - k_x x - k_y y)] \qquad (4 \cdot 7)$$

where ω is the circular frequency and $k_{(x,y)}$ are the (x,y)-directional wave numbers with $j = \sqrt{-1}$. Substitution of (4.7) into (4.4) yields

$$\tilde{\mathbf{L}}(k_x, k_y, \omega) = \begin{bmatrix} -Dk_x^2 - D_{11}k_y^2 - D_s + I\omega^2 & -D_{12}k_x k_y & -jD_s k_x \\ & -Dk_y^2 - D_{11}k_x^2 - D_s + I\omega^2 & -jD_s k_y \\ sym. & & -D_s\tilde{\nabla}^2 - m\omega^2 \end{bmatrix} \qquad (4 \cdot 8)$$

with the corresponding uncoupled continuum Fourier operator given by

$$\tilde{l}^C = \left[(D\tilde{\nabla}^2 + I\omega^2)(\tilde{\nabla}^2 + \frac{m}{D_s}\omega^2) - m\omega^2 \right] \qquad (4 \cdot 9)$$

where the Fourier-transformed ∇^2 operator is defined as

$$\tilde{\nabla}^2 = -(k_x^2 + k_y^2) \qquad (4 \cdot 10)$$

The Fourier-transformed continuum matrix operator, $\tilde{\mathbf{L}}(k_x, k_y, \omega)$, given by (4.8) and the uncoupled continuum operator, $\tilde{l}^C$, by (4.9) will serve as our reference equations with which the corresponding *discrete* operators and characteristic equations will be compared in order to assess the effect of mass lumping on plate vibrations.

The general discrete Fourier operator that approximates the continuum case by the four-noded plate bending element can be shown to be (Park and Flaggs, 1985):

$$\bar{\mathbf{L}}^G =$$

$$\begin{bmatrix} -D\bar{k}_x^2\bar{l}_y & -D_{12}\bar{k}_x\bar{k}_y\sqrt{\chi_x\chi_y} & -jD_s\bar{k}_x\sqrt{\chi_x}\bar{l}_y \\ -D_{11}\bar{k}_y^2\chi_x & & \\ -D_s\chi_x\bar{l}_y & & \\ +I\omega^2[(1-\alpha) & & \\ +\bar{l}_x\bar{l}_y] & & \\ & -D\bar{k}_y^2\bar{l}_x & -jD_s\bar{k}_y\sqrt{\chi_y}\bar{l}_x \\ & -D_{11}\bar{k}_x^2\chi_y & \\ & -D_s\chi_y\bar{l}_x & \\ & +I\omega^2[(1-\alpha) & \\ & +\bar{l}_x\bar{l}_y] & \\ sym. & & D_s(\bar{k}_x^2\bar{l}_y+\bar{k}_y^2\bar{l}_x) \\ & & -m\bar{l}_x\bar{l}_y\omega^2 \end{bmatrix} \quad (4\cdot 11)$$

where the discrete fourier numbers, $\bar{k}_x$ and $\bar{k}_y$, and the so-called directional averaging operators, $\bar{l}_{(x,y)}$ and $\chi_{(x,y)}$, are given by

$$\begin{cases} \bar{k}_x = \sin(k_x l_x/2)/l_x/2 \\ \chi_{(x,y)} = 1 - \frac{l^2_{(x,y)}}{4}\bar{k}^2_{(x,y)} \\ \bar{l}_{(x,y)} = 1 - \frac{l^2_{(x,y)}}{6}\bar{k}^2_{(x,y)} \end{cases} \quad (4\cdot 12)$$

and α is a coefficient defined in the mass matrix modeling formula (2.16).

The relation between the frequency vs. the wave number for the continuum and the discrete cases can be obtained by requiring the determinant of (4.8) and (4.11) to be zero. Figure 4 shows the normalized frequency ($\omega\ell^2\sqrt{\rho/D}$) vs. the wave number ($k\ell$) for the continuum and the discrete cases with various mass matrix choices based on (2.16). It is noted that the two cases correspond to an infinite plate or so-called interior solutions. Judging from Fig. 4, one may conclude that the choice of $\alpha = 0.5$ (the average of the lumped and the consistent matrices) should perform best for plate vibration analysis. For plates with finite dimensions as we shall see, the influence of boundary edges must be taken into consideration in utilizing the dispersion curve for selecting an appropriate choice of mass matrix.

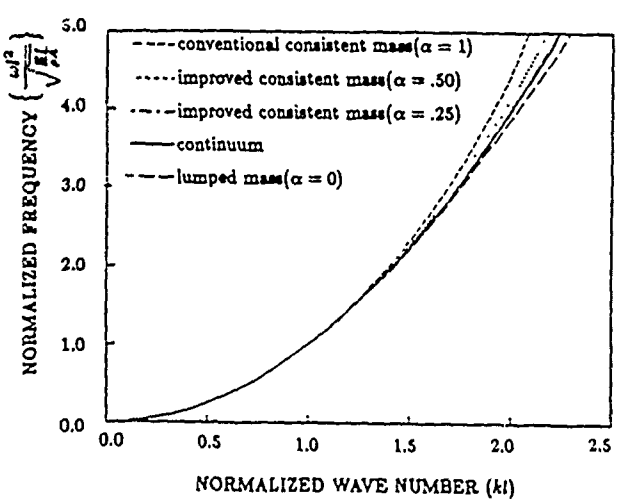


Figure 4. Dispersion Curve for Plate ($\nu = 0.343$)

Table 2 summarizes the vibration analysis results with different mass matrices for a plate with free edges. In computing the errors in frequency computations by the four-noded element, we have assumed that the frequencies given in (Leissa, 1969) to be the converged answers. The finite element analysis results indicate that the discrete frequency vs. wave number curve shown in Fig. 4 overestimates the frequency error from the above. We conjecture that this overestimation manifested in the present discrete Fourier analysis may have been due to the failure of the four-node element to satisfy rigorously the free-edge conditions.

Table 2. Frequency Errors Computed by Different Mass Models for Free-Free Plate

MASS TYPE	MODE	4x4 Mesh	16x16 Mesh	36x36 Mesh
Lumped Mass $\alpha = 0.0$	1	-30.1	-8.5	-3.9
	2	-32.8	-15.7	-8.4
	3	-25.5	-11.2	-5.7
	4	-	19.4	-10.0
	5	-	-	-20.6
$\alpha = .25$	1	-24.7	-6.2	-2.7
	2	-26.4	-11.5	-6.0
	3	-17.7	-7.0	-3.4
	4	-	14.7	-7.2
	5	-	-	-17.0
$\alpha = .50$	1	-17.7	-3.6	-1.5
	2	-17.7	-6.8	-3.4
	3	-8.0	-2.1	-0.9
	4	-	9.1	-4.2
	5	-	-	-12.9
$\alpha = .75$	1	-8.5	-0.7	-0.2
	2	-5.1	-1.2	-0.7
	3	6.1	3.7	1.8
	4	-	2.2	-0.8
	5	-	-	8.2
Consistent Mass $\alpha = 1.0$	1	4.8	2.4	1.1
	2	16.1	5.6	2.4
	3	29.8	10.6	4.8
	4	-	6.5	2.8
	5	-	-	2.5

The mode-by-mode convergence characteristics for the first three modes are shown in Figs. 5-7. With the exception of the first mode with a (2×2)-mesh, it is observed that the choice of $\alpha = 3/4$ consistently yields the most accurate results.

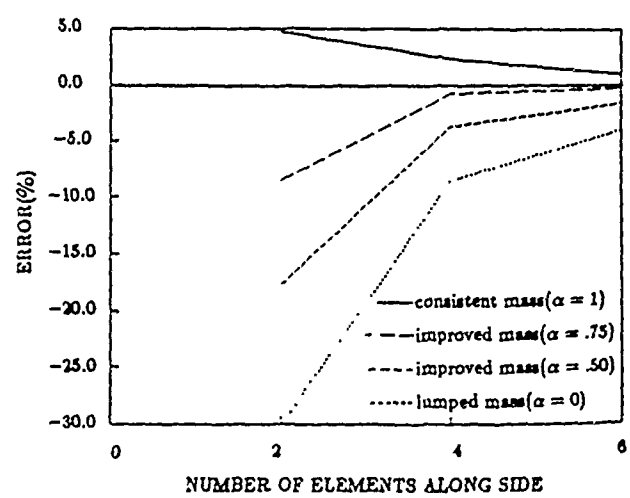


Figure 5. Error in First Mode for Different Mass Modeling

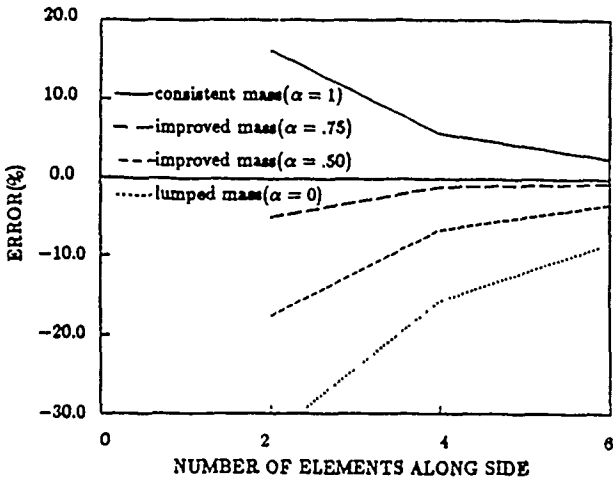


Figure 6. Error in Second Mode for Different Mass Modeling

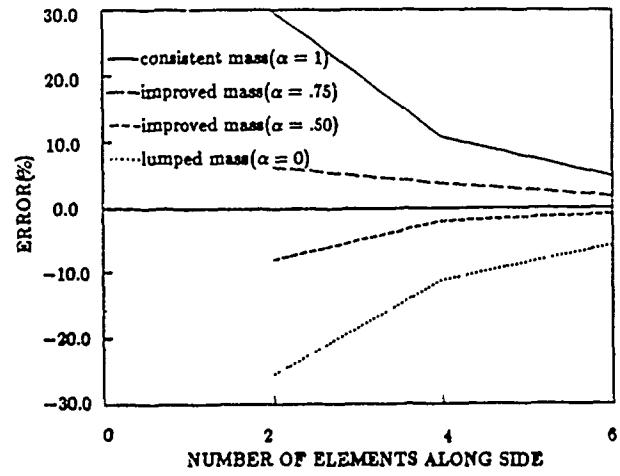


Figure 7. Error in Third Mode for Different Mass Modeling

5. Discussions

In the present paper a systematic technique of obtaining the lumped mass matrices for vibration and transient analyses has been proposed. The technique is based on the symbolic discrete Fourier analysis and reduces to the known mass lumping techniques for simple elements.

In order to further increase the accuracy of frequency computations, an improved form of mass matrix is proposed, which is a combination of the lumped and consistent mass matrices. Numerical results conducted for a plate with free edges indicate that substantial improvements in intermediate frequency computations can be realized if one judiciously employs a combined mass matrix.

So far the present work has focused on modifying the mass matrices in order to improve the frequency accuracy. Our future work will focus on the combined tailoring of both the mass and stiffness matrices to capture inermediate frequency components more accurately. It should be noted that the accuracy of intermediate frequency components based on the finite element methods is increasingly important in the areas of control of flexible structures, underwater acoustics and wave propagation through composites

Acknowledgements.

The work reported herein was partially supported by Naval Research Laboratory under Contract N00014-87-K-2118. We thank Ms. Louise Schuetz of NRL for her interest and encouragement. We also thank Prof. Charbel Farhat for his assistance in our symbolic eigenvalue analysis using MACSYMA.

References

1. Archer, J. S., (1963), "Consistent Mass Matrix for Distributed Mass Systems," *J. of the Structural Division*, **ST 4**, pp. 161-178,

2. Bažant, Z. P., (1978), "Spurious Reflection of elastic waves in non-uniform finite element grids," *Computer Methods in Applied Mechanics and Engineering*, **16**, 91-100.

3. Bažant, Z. P. and Celep, C., (1982), "Spurious Reflection of Elastic Waves in Nonuniform Meshes of Constant and Linear Strain Elements," *Comp. and Struc.*, **15**, No. 4, pp. 451-459.

4. Belytschko, T. and Mullen, R., (1978), "On Dispersive Properties of Finite Element Solution," *Modern Problems in Wave Propagation*, Edited by J. Miklowitz and J. D. Achenbach, John Wiley and Sons, New York, pp. 67-82.

5. Celep, Z. and Turhan, D., (1987), "Transient Wave Propagation in Constant and Linear Strain Elements, *J. Sound and Vibration*, **116**(1), 15-23.

6. Flaggs, D. L., (1988), *Symbolic Analysis of the Finite Element Method in Structural Mechanics*, PhD Thesis, Stanford University.

7. Fried, I. and Malkus, D. S., (1975), "Finite Element Mass Matrix Lumping by Numerical Integration with No Convergence Rate Loss," *Int. J. Solids Structures*, **11**, pp. 461-466.

8. Hinton, E., Rock, T. and Zienkiewicz, O. C., (1976), "A Note on Mass Lumping and Related Processes in the Finite Element Method," *Earthquake Eng. and Structural Dynam.*, **4**, pp. 245-249.

9. Krieg, R. D. and Key, S. W., (1973), "Transient Shell Response by Numerical Time Integration," *Int. J. Numerical Methods in Eng.*, **7**, pp. 273-286.

10. Leckie, F. A. and Lindberg, G. M., (1963), "The Effect of Lumped Parameters on Beam Frequencies," *Aeronaut. Quarterly*, **14**, pp. 224-240.

11. Leissa, A. W., *Vibration of Plates*, NASA SP-160, Washington, D. C., 1969, 87-114.

12. Park, K. C. and Flaggs, D. L., (1984), "A Fourier Analysis of Spurious Mechanism and Locking in the Finite Element Method," *Comp. Meth. in Appl. Mech and Engng.*, **46**, pp. 68-81.

13. Park, K. C. and D. L. Flaggs, (1985), "A Symbolic Fourier Synthesis of a One-Point Integrated Quadrilateral Plate Element," *Comp. Meth. Appl. Mech. Engr.*, **48**, 203-236.

14. Rock, T. and Hinton, E., (1974), "Free Vibration and Transient Response of Thick and Thin Plates Using the Finite Element Method," *Int. J. Earthq. Engng. Struct. Dyn.*, **3**, pp. 51-63.

15. Tong, P., Pian, T. H. H. and Buciarelli, L. L., (1971), "Mode Shapes and Frequencies by the Finite Element Method Using Consistent and Lumped Mass," *J. Comp. Struct.*, **1**, pp. 623-638.

16. Vichnevetsky, R. and Bowles, J. B., (1982), *Fourier Analysis of Numerical Approximations of Hyperbolic Equations*, SIAM Studies in Applied Mathematics, Philadelphia, Pa.

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING, VOL. 30, 27–55 (1990)

TRANSIENT FINITE ELEMENT COMPUTATIONS ON 65536 PROCESSORS: THE CONNECTION MACHINE

C. FARHAT, N. SOBH AND K. C. PARK

Department of Aerospace Engineering Sciences and Center for Space Structures and Controls, University of Colorado at Boulder, Boulder, CO 80309-0429, U.S.A.

SUMMARY

This paper reports on our experience in solving large-scale finite element transient problems on the Connection Machine. We begin with an overview of this massively parallel processor and emphasize the features which are most relevant to finite element computations. These include virtual processors, parallel disk I/O and parallel scientific visualization capabilities. We introduce a distributed data structure and discuss a strategy for mapping thousands of processors onto a discretized structure. The combination of the parallel data structure with the virtual processor mapping algorithm is shown to play a pivotal role in efficiently achieving massively parallel explicit computations on irregular and hybrid two- and three-dimensional finite element meshes. The finite element kernels written in C* / Paris have run with success to solve several examples of linear and non-linear dynamic simulations of large problem sizes. From these example runs, we have been able to assess in detail their performance on the Connection Machine. We show that mesh irregularities induce an MIMD (Multiple Instruction Multiple Data) style of programming which impacts negatively the performance of this SIMD (Single Instruction Multiple Data) machine. Finally, we address some important theoretical and implementational issues that will materially advance the application ranges of finite element computations on this highly parallel processor.

1. INTRODUCTION

Parallel computers are having a profound impact on computational mechanics. This is reflected by the continuously increasing number of publications on finite elements and parallel processing. Not only have some computational strategies been re-designed for implementation on commercially available multiprocessors, but also some innovative algorithms have been spurred by the advent of these new machines. However, many of the reported parallel finite element simulations have been on systems with a few processors. Examples of these systems are Intel's iPSC with 32 processors (reported by Farhat and Wilson[1]), JPL/Caltech's hypercube with 32 processors,[2] Alliant's FX8 model with 8 processors[3,4] and CRAY's systems with up to 4 processors.[5] (For more complete lists of references on this topic see White and Abel[6] and Noor.[7] While great speed-ups were measured on these coarse to medium grain machines, Farhat[8] has shown that traditional vector supercomputers could not be outperformed in finite element simulations (except of course on systems which connect more than one vector superprocessor, such as the CRAY X-MP and CRAY-2 systems, each of which has 4).

Recently, massively parallel machines have demonstrated their potential to be the fastest supercomputers, a trend that may accelerate in the future. While solving the shallow water equations, McBryan has reported that the Connection Machine (CM_2 in the sequel) (65 536 processors) was three times faster than the four-processor CRAY X-MP.[9] Gustafson *et al.* have

0029-5981/90/090027-29$14.50
© 1990 by John Wiley & Sons, Ltd.
Received 11 March 1989
Revised 12 September 1989

developed highly parallel solutions for baffled surface wave equations, unstable fluid flow and beam strain analysis, and have reported performances on NCUBE's 1024-processor hypercube which are close to those of vector supercomputers.[10]

The objective of the present study has been: first, to evaluate the multiprocessing features of the CM-2 that are relevant to finite element computations; second, to develop a suitable finite element data structure which exploits the system architecture; third, to implement a decomposition/mapping procedure that matches as far as possible the layout of the processors to the finite element meshes; and fourth, to assess those implications of finite element analysis on the CM-2 that should be considered in the design of future massively parallel processors. Hence, we focus primarily on implementational issues that are critical for the full exploration of the multiprocessing capabilities of the CM-2, and only secondarily on solution algorithms, as far as they impact the present study on implementational issues.

The finite element equations of motion for structural systems can be expressed as

$$\mathbf{M}\ddot{\mathbf{d}} + \mathbf{F}^{in}(\dot{\mathbf{d}}, \mathbf{d}) = \mathbf{F}^{ex} \tag{1}$$

where $\mathbf{M}$ denotes the positive definite lumped mass matrix, $\mathbf{F}^{in}$ and $\mathbf{F}^{ex}$ denote the internal and external force vectors, and $\ddot{\mathbf{d}}$, $\dot{\mathbf{d}}$ and $\mathbf{d}$ denote respectively the acceleration, velocity and displacement vectors. In the linear case, the internal force vector becomes

$$\mathbf{F}^{in} = \mathbf{D}\dot{\mathbf{d}} + \mathbf{K}\mathbf{d} \tag{2}$$

where $\mathbf{D}$ and $\mathbf{K}$ are the damping and stiffness matrices respectively, which are positive semi definite. In this work, an eventual damping is assumed to be proportional to the mass and stiffness.

The algorithmic nature of a candidate solution method for the structural dynamics equation (1) can significantly influence the software requirements, data communications and arithmetic efficiency. As our main focus is on implementational issues rather than algorithmic ones, we have decided on a simple explicit time integration procedure. Hence, we choose to integrate equation (1) with the fixed step explicit central difference algorithm because (a) it is inherently parallel, and (b) it has the largest undamped stability limit among second-order accurate explicit linear multistep algorithms, as has been demonstrated by Krieg.[11] In our context, it expressed as

$$\begin{aligned} \dot{\mathbf{d}}^{n+1/2} &= \dot{\mathbf{d}}^{n-1/2} + h\mathbf{M}^{-1}(\mathbf{F}^{ex}(t^n) - \mathbf{F}^{in}(\dot{\mathbf{d}}^n, \mathbf{d}^n)) \\ \mathbf{d}^{n+1} &= \mathbf{d}^n + h\dot{\mathbf{d}}^{n+1/2} \end{aligned} \tag{3}$$

where h is the fixed time step and the superscript n indicates the value at the discrete time t^n.

The remainder of this paper deals with the massively parallel solution of (1) using (3), and is organized as follows. In Section 2, we give an overview of the CM-2 hardware configuration and emphasize those features which are pertinent to finite element computations. In particular, we address issues that are related to the processor memory size, to the SIMD architecture and to the fast interprocessor communication package, the *NEWS grid*. In Section 3, we discuss the floating point arithmetic performance of the CM-2 and highlight its current dependence on the selected language compiler. Algebraic manipulations coded in *Lisp are shown to be three times as fast as when written in C*. A general purpose finite element distributed data structure is presented in Section 4. Designed originally to handle massively parallel finite element explicit computations on irregular and hybrid meshes, this parallel data structure is also very efficient for parallel I/O manipulations and parallel graphic animation. Since the often-encountered mesh irregularities inhibit the use of the *NEWS grid* communication package, we discuss in Section 5 an alternative decomposition, mapping strategy. The decomposition technique is designed to minimize both the

amount of communication between different chips and the amount of wire contention within a chip. The mapping algorithm attempts to reduce the distance that information must travel. Section 6 summarizes the overall organization of the massively parallel transient simulation. In Section 7, our parallel data structure and processor mapping are applied to (3) for the solution of various large-scale transient problems. Measured performances are analysed in detail. Mesh irregularities are shown to be the source of several factors which considerably slow down the machine. Finally, in Section 8, we address some important theoretical and implementational issues that will materially advance the application ranges of finite element computations on the CM_2. In particular, we note that time integration numerical algorithms such as explicit finite differences and equation solvers such as the preconditioned conjugate gradient are implemented using the same parallel data structure and mapping algorithm which are presented in this paper. We compare the substructuring technique and the virtual processor approach, and comment on the implications of implicit algorithms for the effective use of the CM_2.

2. THE CONNECTION MACHINE HARDWARE ARCHITECTURE

Here we present an overview of the CM_2 system organization and discuss issues that are pertinent to massively parallel finite element computations. See Hillis[12] for an in-depth discussion on the rationale behind the CM_1 (a previous model of the Connection Machine), the Technical Summary of Thinking Machines Corporation[13] for further architectural information and McBryan[9] for initial studies of scientific computations on the CM_1. For the sake of clarity, we summarize the architectural features before discussing their impact on finite element simulations.

2.1 System organization

2.1.1. CM_2: The parallel processing unit. The CM_2 is a cube 1·5 m on a side, made of up to eight subcubes (Plate 1). Each subcube contains 512 chips and every chip includes 16 bit serial processors which are connected by a switch. Each individual processor has 64 Kbits (8 Kbytes) of bit-addressable local memory and an arithmetic-logic unit (ALU) that can operate on variable-length operands. Every two chips may share an optional Weitek floating point accelerator chip. A fully configured CM_2 thus has 4096 (2^{12}) chips, 2048 floating point accelerator chips, 65 536 processors and 512 Mbytes of memory. The chips are arranged in a 12-dimensional hypercube. A chip i is directly connected to 12 other chips j, with the binary representation of i and j differing only by 1 bit. The CM_2 system provides two forms of communication between the processors.

- A general mechanism known as the *router* which allows any processor to communicate with any other processor. Each CM_2 chip contains one router node i which serves the 16 processors on the chip, numbered $16i$ to $16i + 15$. The router nodes on all the chips are wired together in a 12-dimensional Boolean cube and together form the complete router network (Figure 1). For example, suppose that processor 117 (processor 5 on router node 7) has a message to send to processor 361 (processor 9 on router node 22). Since $22 = 7 + 2^4 - 2^0$, router 7 forwards the message to router 6 ($6 = 7 - 2^0$) which forwards it to router 22 ($6 + 2^4$), which delivers the message to processor 361.
- A more structured and somewhat faster communication mechanism called the *NEWS grid*. Each processor is wired to its four nearest neighbours in a two-dimensional rectangular grid (Figure 2). Communication on the *NEWS grid* is extremely fast and recommended whenever it is possible.

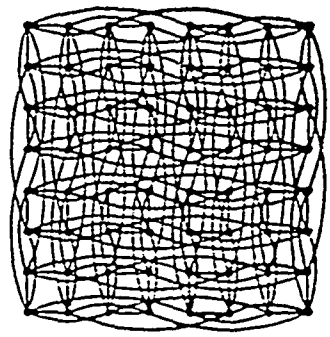


Figure 1. The router network

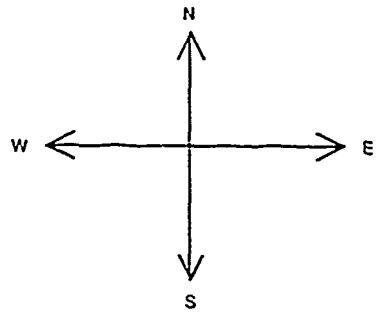


Figure 2. The NEWS grid

An important practical feature of the CM_2 is the support for *virtual* processors. When the CM_2 is initialized for a run, the number of virtual processors (vp in the sequel) may be specified. If it exceeds the number of available physical processors, then the local memory of each processor is split up into a number of regions equal to the ratio between the number of vps and the number of physical processors. Automatically, for every Paris (PARallel Instruction Set) instruction, the processors are time-sliced among the regions. If a physical processor is simulating N vps, each Paris instruction is decoded by the sequencer (as explained below) only once for N executions. This results in an enhanced *user* performance. Also, the use of a vp > 1 allows the pipelining of floating point operations in the Weitek chips, which provides an additional enhancement to *machine* performance. The system organization of a CM_2 is shown in Figure 3.

The CM_2 is an SIMD machine. All processors must execute identical instructions or some processors may choose to ignore any instruction. Consequently, an instruction which involves a nested binary branch can see its execution time increased by a factor of two. The SIMD nature of the CM_2 has some disadvantages in finite element computations, as will be shown.

2.2. *Impact on finite element computations*

It is well known that the solution algorithm (3) can be implemented using only element-level computations. Hence, if each vp of the CM_2 is mapped onto one finite element, equation (1) can be efficiently integrated in parallel. The rationale behind this processor-to-element assignment

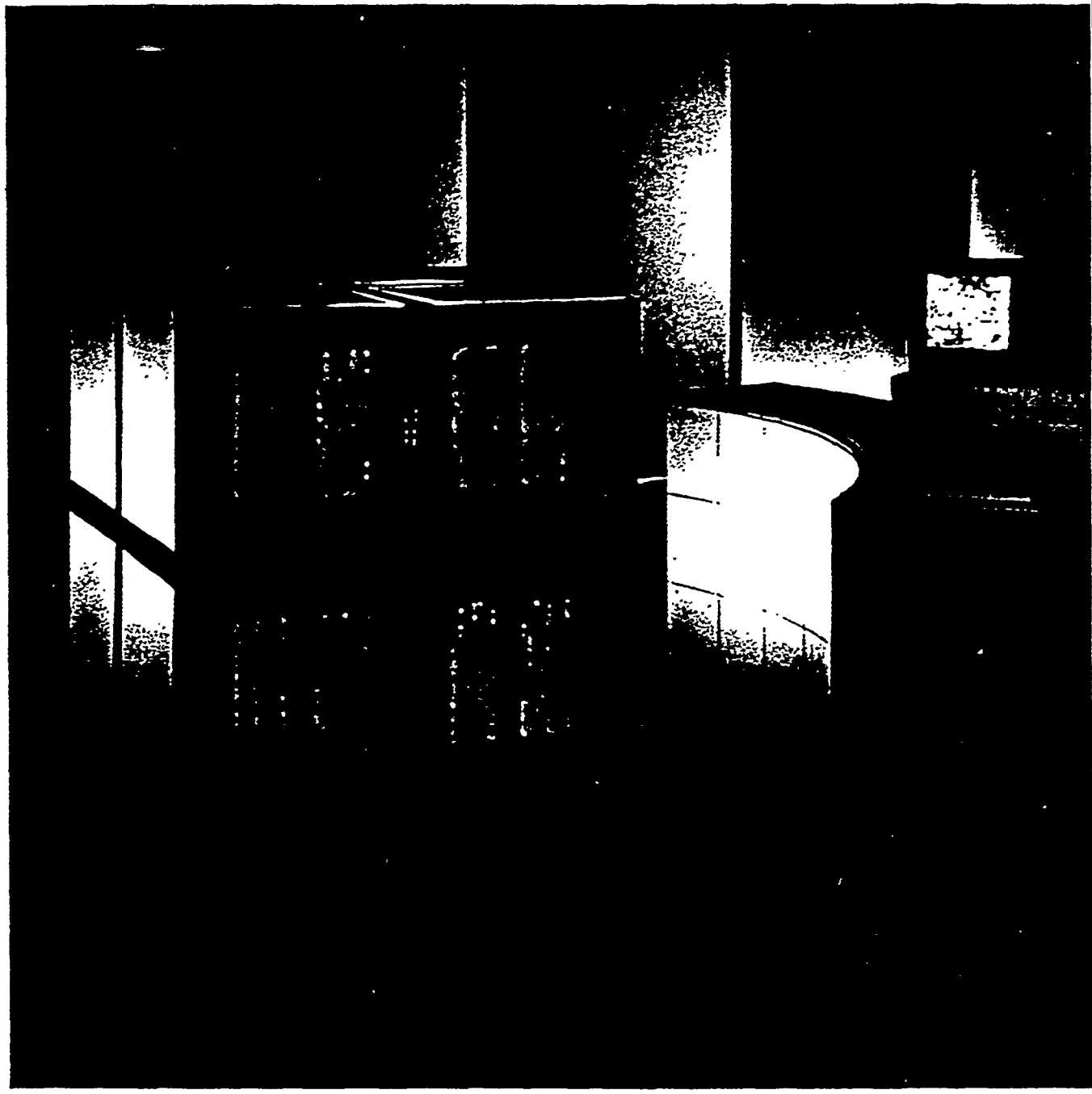


Plate 1. The CM 2

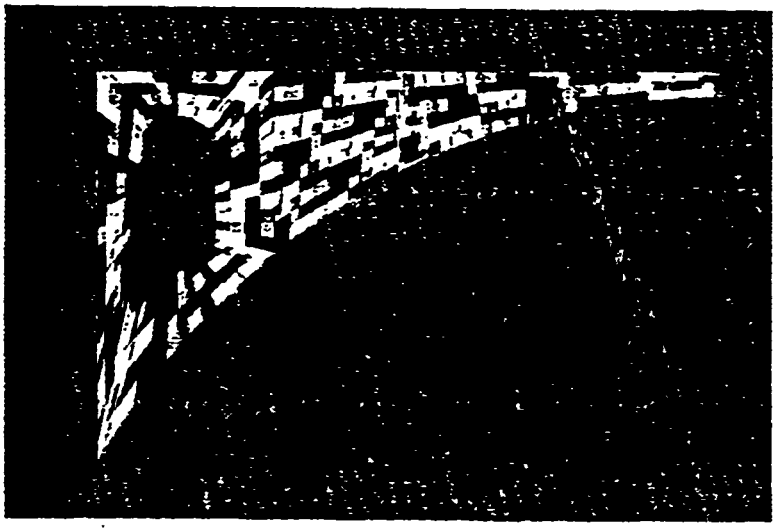


Plate 2. Discretization and decomposition of a tapered beam

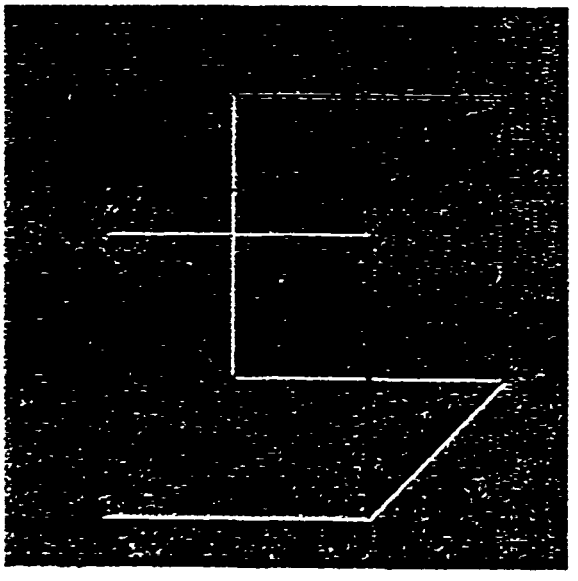


Plate 3. Irregular cell

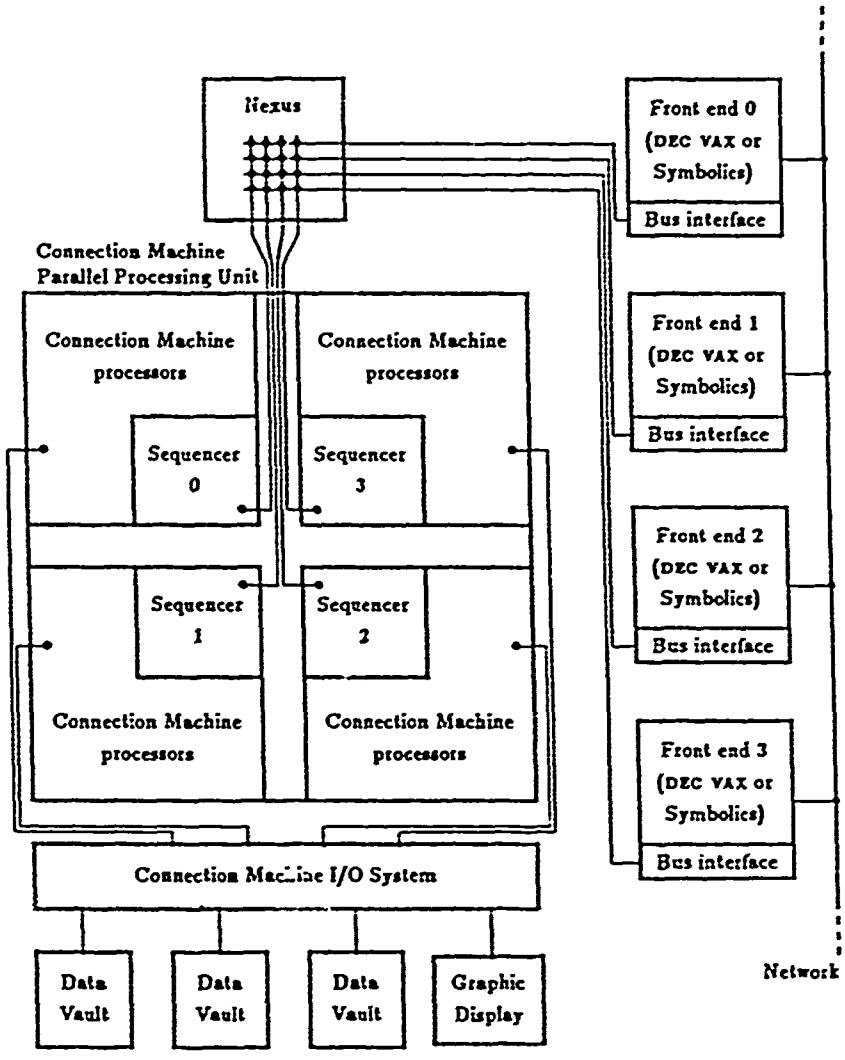


Figure 3. System organization of a CM-2

will be analysed in Sections 4 and 8. Here, we discuss the direct impact of the CM-2 hardware on such a decision.

2.2.1. The local memory and element level computations. Consider the 9-node curved shell element shown in Figure 4. Three displacements and two rotations are attributed to each node, which amounts to a total of 45 degrees of freedom per element. Consequently, the symmetric part of the elemental stiffness matrix, $K^{(e)}$, contains $45*(45+1)/2 = 1035$ words. If double precision is used, the storage of $K^{(e)}$ amounts to $1035*64 = 66\,240$ bits, which exceeds the 65 536 bits that are available on a single CM-2 processor. On the other hand, if single precision is used, the storage of $K^{(e)}$ requires 33 120 bits, so that 32 416 bits are left for the storage of the vectors $d^{(e)}$, $\dot{d}^{(e)}$, the elemental lumped mass vector $M^{(e)}$ and the forces $F^{ext(e)}$ and $F^{int(e)}$. However, even in the latter

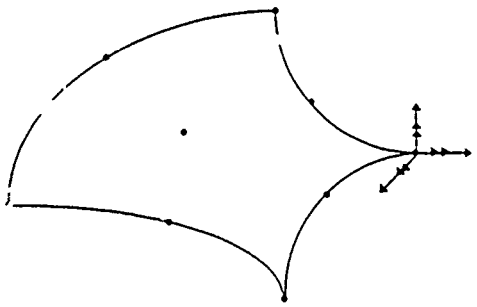


Figure 4. A 9-node shell element

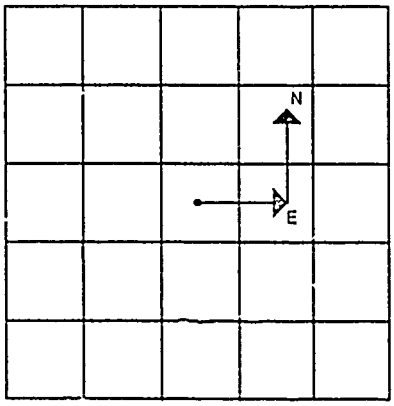


Figure 5. A two-step *NEWS* mechanism on a regular mesh

case, only a vp ratio of 1 can be used. This limits the size of the finite element mesh to the maximum number of processors available on the CM_2 at hand. Also, it inhibits further performance enhancement, as outlined in Section 2.1.

Fortunately, in our case the above storage requirements can be considerably decreased. The nature of explicit computations is such that $\mathbf{F}^{int}(\mathbf{d}^n)$ can be directly computed from the displacements at t^n and the stress–strain constitutive equation. As a result, the solution process defined in (3) involves only vector quantities which do not require a large amount of storage, so that vp ratios between 1 and 4 are possible. However, the reader should keep in mind that the current local memory size of a CM_2 processor may penalize sophisticated high order elements and implicit finite element algorithms in general. This restriction is not encountered on other commercially available hypercubes such as iPSC, NCUBE and AMETEK among others.

2.2.2. The NEWS grid and finite element patches. Consider the regular finite element mesh shown in Figure 5. Except on the boundaries, each element is connected in the same pattern to exactly eight other elements. Consequently, during the explicit time integration algorithm, each processor communicates with its neighbours in the same manner. Interprocessor communication can be performed with a two-step *NEWS* mechanism (Figure 5). However, the beauty of the finite element method resides in the fact that it solves models with irregular meshes. Typically, a finite

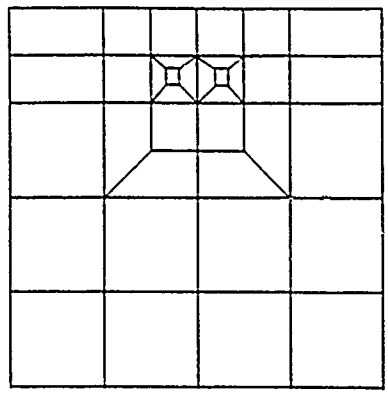


Figure 6. Transition zones

element mesh consists of several patches which are connected together using irregular transition regions (Figure 6). For these often encountered cases, the *NEWS grid* becomes impractical. Rather, the *router* has to be utilized. In Section 4, we describe how a distributed data structure can guide the router during this process.

2.2.3. SIMD hardware vs. MIMD finite element computations. Typical finite element meshes comprise more than one type of element. Consider the case where a discretized region is modelled with shell elements that are stiffened with beam elements. Clearly, the instructions associated with the shell elements differ from those associated with the beam elements. Consequently, the vps which are assigned to shell elements and the vps which are assigned to beam elements cannot execute their segments of code in parallel; for example, the beam processors have to execute first, then the shell processors. If T_b and T_s denote the execution times associated with the instructions for a beam and a shell element respectively, the total elapsed parallel time for a single instruction over the set (beams + shells) on an SIMD multiprocessor is $T_b + T_s$. On an MIMD multiprocessor, this elapsed parallel time is $\max(T_b + T_s)$. Similar situations arise when during the loading some elements turn to be materially non-linear and some remain linear. In this case, one should always compute the linear component of the response (the elastic stiffness for example) before attempting to test the yielding criterion. However, in spite of these disadvantages SIMD programs can still be attractive, because they tend to be easier to debug and rarely suffer from the synchronization errors which are typical of MIMD codes.

2.2.4. Parallel I/O in finite element computations. At each time step, the computed displacements, velocities, accelerations as well as strains and stresses need to be stored on disks. This represents a significant amount of I/O traffic. It has been our experience that the CM-2 Data Vault system is efficient at reducing the corresponding elapsed time (see Section 7).

2.2.5. Real-time graphics animations. The massively parallel real-time animation of the mesh deformations is a direct consequence of the availability of the *Frame Buffer* and decision of assigning a vp to a finite element. At each time step, after the node displacements are found all of the vps concurrently draw the outline of their assigned elements on the graphic screen. The result is a real-time finite element animation.

3. BENCHMARKING THE CM_2

At the time of writing this paper, the CM_2 supports three high level languages: C* (pronounced see-star), *Lisp (pronounced star-lisp) and CM-Lisp (pronounced see-m-lisp). The first two are extensions of C and Lisp respectively. Paris is somewhat the assembly language of this parallel processor.

In this section we comment on the results of a set of timing experiments that were carried out on the CM_2 of the Center for Applied Parallel Processing (CAPP), at the University of Colorado, Boulder. Since only one eighth of a cube was available on this system, all results were obtained using 8192 processors. McBryan[9] has shown that all results demonstrated on subcubes of the CM_2 scale essentially linearly to the 65 536 processor system. Consequently, throughout this paper, megaflop rates are reported after they are linearly scaled to the full configuration. These experiments provided us with:

- a reference performance for the evaluation of our approach to massively parallel finite element explicit computations
- the influence of the vp ratio and that of the high level language compiler on attainable performances. At this point, we remind the reader that, if an application requires an amou[illegible] of local memory (per processor) m_a, the highest vp ratio possible is equal to the closest p[illegible] of two to the ratio between the maximum amount of local memory available on the m[illegible] (currently 8 Kbytes), and m_a.

Table I reports the megaflop rates for some scientific computations on the CM_2 at different vp ratios. All statements were written in C*. Each statement is performed by each processor on its variables. All variables were declared parallel (local) and float (simple precision), except variable *dp* which was declared mono (serial) float, and variable *i* which was declared mono integer. Timings were measured using the *cmtimer* routines. Each '+' operation or '*' operation was counted as one flop.

Based on these results, we have observed the following:

1. Floating point performance is enhanced at higher vp ratios. This is due to the fact that for vp ratios greater than one, computations in the Weitek chip are pipelined.
2. Vector saxpys are not slower than scalar ones. This is because memory addresses are computed on the front end. The additional speed noticed for vector saxpys is thought to be due to the overlapping of addressing and floating point computations.
3. C* appears to handle poly (parallel) assignments poorly. This can be seen by comparing the performances of the dot product and the vector multiply. Each of these two vector

Table I. Megaflop rates using C*

Parallel processor = CM_2—Language = C*—Variable = float									
Statement	vp ratio								
	1	2	4	8	16	32	64	128	256
$y[i] += x*x[i]$	740	808	848	850	880	—	—	—	—
$y = y + x*x$	569	654	699	728	743	761	778	791	800
$z = x*y$	409	485	535	569	579	585	600	610	623
$dp += x*y$	202	359	583	839	1075	1240	1348	1400	1500

operations requires one floating point per processor. In addition, the dot product requires a reduction (accumulation phase) which necessitates communication. However, at high vp ratios, the dot product is twice as fast as the vector multiply! (At low vp ratios, the amount of floating point computations is not large enough to amortize the price of communication.) Since the dot product does not store any value in the processor memory and the vector multiply stores the result of $x * y$ back into z, this leads us to believe that the C* compiler generates a code which is very inefficient at handling assignments. This also explains why the saxpy exhibits a higher megaflop rate than the vector multiply: it has twice as many floating point computations for one assignment.

The same computations were repeated using *Lisp. The comparison of both sets of timings for the maximum vp demonstrates a formidable superiority of the *Lisp compiler (see Figure 7). This is partly due to the fact that it has been used longer on the CM_2 than C*. In spite of the proven superior efficiency of *Lisp over C*, we have chosen to implement our finite element code using C* because of our familiarity with C.

4. FINITE ELEMENT PARALLEL DATA STRUCTURES

Consider again the explicit central difference algorithm:

$$\begin{aligned} \dot{\mathbf{d}}^{n+1/2} &= \dot{\mathbf{d}}^{n-1/2} + h\mathbf{M}^{-1}(\mathbf{F}^{ex}(t^n) - \mathbf{F}^{in}(\dot{\mathbf{d}}^n, \mathbf{d}^n)) \\ \mathbf{d}^{n+1} &= \mathbf{d}^n + h\dot{\mathbf{d}}^{n+1/2} \end{aligned} \tag{4}$$

The global mass matrix M is assembled once. At each time step t^n, the computations are dominated by the evaluation of the internal forces:

$$\mathbf{F}^{in} = \sum_{e=1}^{e=n_{el}} \int_{\Omega^{(e)}} [LS]^T \sigma \, d\Omega$$

where σ is the stress vector, S are the shape functions, L is a partial derivative operator and $\Omega^{(e)}$ is the area of the eth finite element. Clearly, the parallel computation of $\mathbf{F}^{in}$ is best done element-by-element. Thus, equation (1) can be efficiently integrated in parallel if the CM_2 virtual processors are mapped onto the elements of the mesh. This is a departure from the grid point massively parallel computations advocated by Thinking Machines Corporation for the CM_2.[13] First, all processors compute concurrently the local forces $\mathbf{F}^{ex(e)}(t^n)$ and $\mathbf{F}^{in(e)}(\dot{\mathbf{d}}^n, \mathbf{d}^n)$. Next, these contributions are accumulated through communications among processors that are mapped onto neighbouring elements.

In this section, we describe the finite element data structures which we have selected to drive the massively parallel computations on the CM_2. These are element oriented, while similar data structures proposed for other hypercubes are subdomain oriented (see Farhat *et al.*[14] and Fox *et al.*[15]). In Section 8, we give further comments on this difference. We group these data structures into two sets.

The first set of data structures deals with element-level parallel computations. To be able to perform locally its assigned element-level computations—that is, to perform these computations without interacting with the front-end machine—each processor must store in its own memory its element type (truss, beam, shell, . . . , number of Gauss points, . . .), its element material properties (density, parameters and coefficients for constitutive equations, damping characteristics, thickness, . . .), its nodal geometry (nodal co-ordinates, number of nodes per element) and its boundary conditions (fixed, free degrees of freedom at each node, prescribed forces at each node).

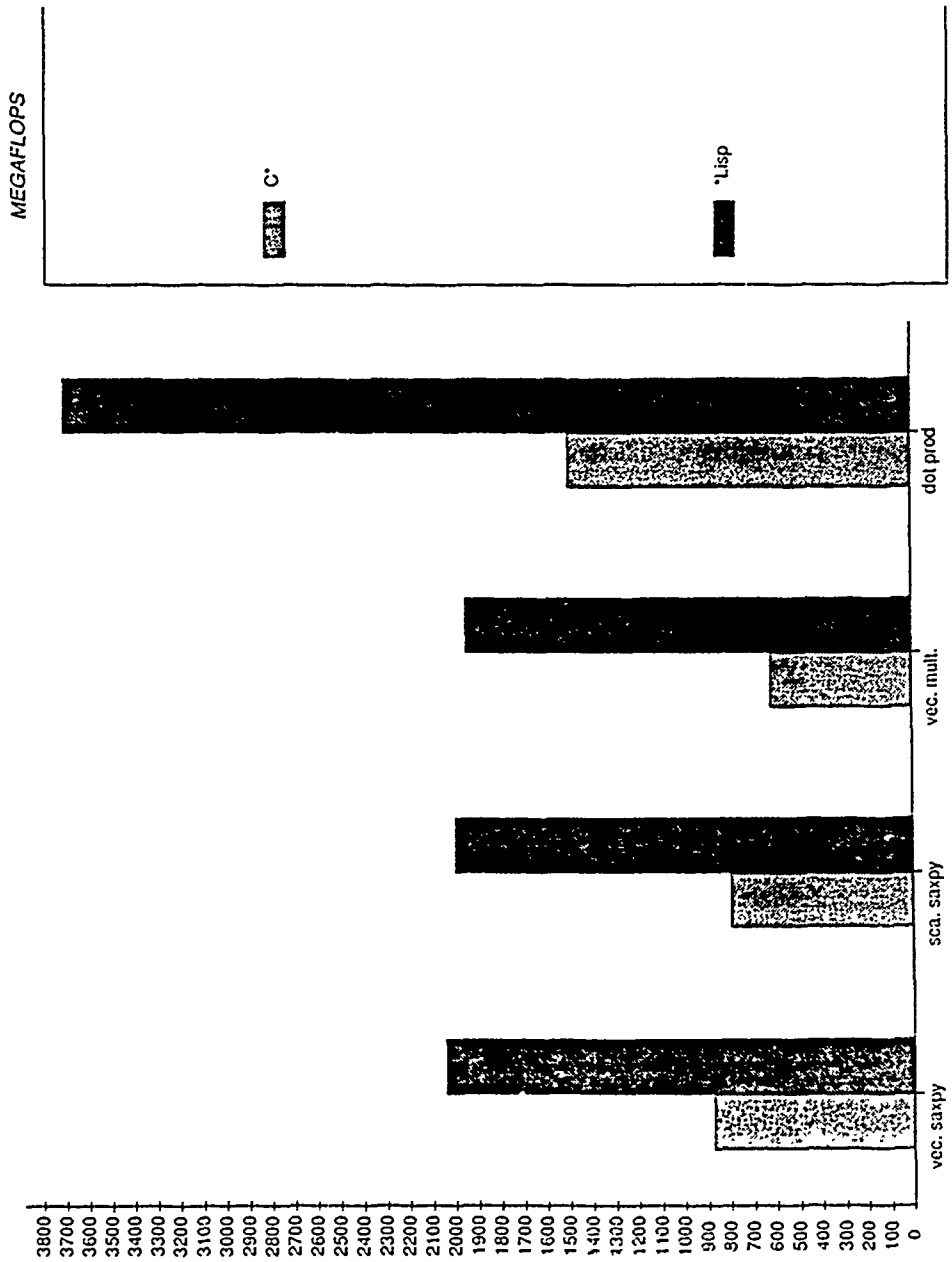


Figure 7. A Comparison of *Lisp and C* performances

This information is compacted in one-dimensional arrays. In addition, each processor must also store in its memory a set of scalars corresponding to computational parameters such as the fixed time step h, and a scalar or one-dimensional buffer for the temporary storage of messages to be passed to neighbouring processors.

The second set of data structures provides the *router* with the mechanism for parallel interprocessor communication. The inability of the *NEWS grid* to handle irregular communication patterns has been addressed in Section 2.2. Let p denote a virtual processor and e_p its assigned finite element. In order to exchange $\mathbf{F}^{\mathrm{in}(e)}(\dot{\mathbf{d}}^n)$ and $\mathbf{F}^{\mathrm{ex}(e)}(t^n)$, virtual processor p must be able to identify at run time:

- the set of processors mapped onto elements adjacent to e_p
- the nodes that e_p shares with these elements
- at each shared node, the degrees of freedom which need to be assembled.

This particular information is vital for meshes with different types of elements. It guarantees that, for example, a moment is not accumulated with a force, or that a force in the x direction is not accumulated with a force in the y direction.

If the above information is gathered in a global form on the front-end machine, most of the execution time which elapses during the accumulation phase would be due to message-passing between the CM_2 processors and the front-end computer. On the other hand, if this information is decentralized—that is, if the memory of processor p is loaded only with the subset of that information which is relevant to the connectivity of e_p—the accumulation phase can be performed without any message-passing between the CM_2 and the front-end computer. Consequently, prior to any computation, the memory of processor p is loaded with the following one-dimensional arrays:

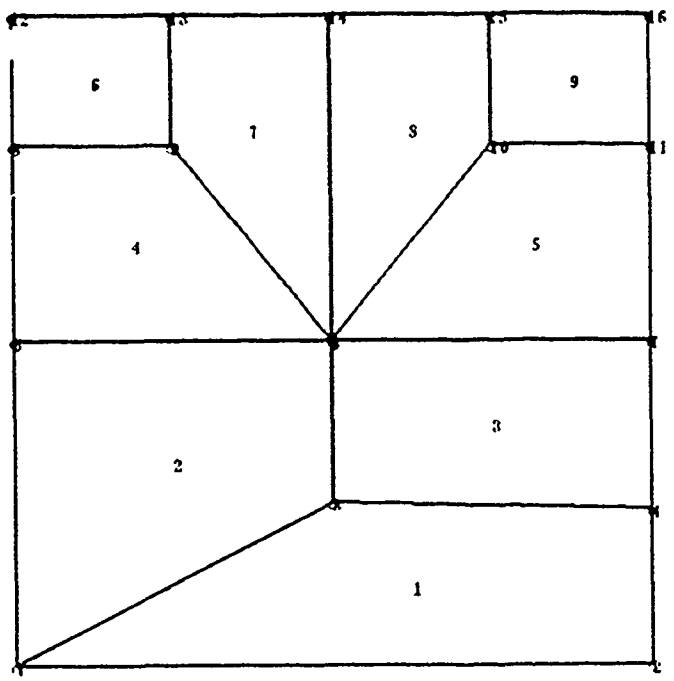
Proc_att_to_node	For each node connected to e_p, it contains the identification of the processors that are mapped onto elements which are also connected to this node. These are stored in a stacked fashion.
Pointer	This is a pointer array. It stores in position i the location in *Proc_att_to_node* of the list of vps that are attached to the node in the ith local position.
Location	For each entry in *Proc_att_to_node*, this array specifies the local position of the shared node in the processor that is mapped onto an element adjacent to e_p

The above arrays are set up by the dedicated finite element mesh analyser which was presented by Farhat *et al.*[14] They require about 80 integer words per processor. Clearly, this is a very small overhead. The mechanism of these arrays is depicted in Figure 8 for element 1. The mesh patch is composed of shell and beam elements.

There is, however, one penalty associated with assigning one element to each vp. The nodes which are common to several elements are duplicated in their corresponding processors. As a result, about 11 per cent of the total memory available on the CM_2 is wasted. This is a small price for the highly parallel computations that are achieved. Given the low cost of memory nowadays, this seems a worthwhile trade off. Moreover, this assignment allows I/O manipulations and graphic post-processing to be trivially parallelized. At each time step, after the nodal displacements are found, all of the processors draw concurrently the outline of their assigned elements on the *Frame Buffer* and send back the results to the front end in parallel.

5. THE DECOMPOSITION/MAPPING STRATEGY

Since the mesh irregularities inhibit the exploitation of the *NEWS grid*, we rely on the data structures of Section 4 to guide the router during interprocessor communication. However, there



Element 1
Proc_att_to_node [2, 3, 3, 2]
Pointer [1, 2, 2, 3, 5]
Location [1, 2, 1, 2]

Figure 8. A distributed data structure for interprocessor communication

is still one additional problem to resolve. Efficiency in massively parallel computations requires the minimization of both the distance that information must travel and, more importantly, the 'hammering' on the router. In the case of finite element computations, this implies that adjacent elements must be assigned, as much as possible, to directly connected processors, and contention for the wire connecting neighbouring chips must be reduced. This defines the mapping problem—that is, it defines which hardware processor is to be mapped onto which finite element of a given mesh.

Farhat[16] developed a heuristic algorithm for mapping massively parallel processors onto finite element graphs and presented some analytical results for corresponding efficiency improvement. Basically, the algorithm searches iteratively for a better mapping candidate through a two-step procedure for the minimization of the communication costs associated with a specific parallel processor topology. Because it seeks a very fast solution for a machine with thousands of processors, this algorithm does not guarantee 'the' optimal mapping. However, it has produced very encouraging results on a variety of non-uniform two and three-dimension.. meshes.

In this work, we adapt the mapping algorithm of Reference 16 to our target parallel processor, the CM_2. The 65 536 processors of this machine are packaged into 4096 16-processor chips, each having its own router node. The 4096 router nodes are arranged in a hypercube of dimension 12. To cope with this topology, we proceed in two steps. First, we decomposes the given mesh into 4096 submeshes, each containing 16 connected finite elements. Next, we apply the mapper given

in Reference 16 to identify which hardware chip is to be mapped onto which submesh. Finally, within each submesh, the elements are numbered randomly between the chip number and the chip number + 15.

Given a finite element mesh, there are several ways to decompose it into 16-element submeshes (see for example Farhat[17] and Malone[18]). Here, each submesh is to be assigned to one chip of the CM_2. In Figure 9, 10 and 11, we show two different decompositions for a discretized square domain, D_1 and D_2.

Both decompositions yield 16 submeshes, each with 16 adjacent elements. Decomposition D_1 was designed to minimize the communication bandwidth—that is, the maximum number of

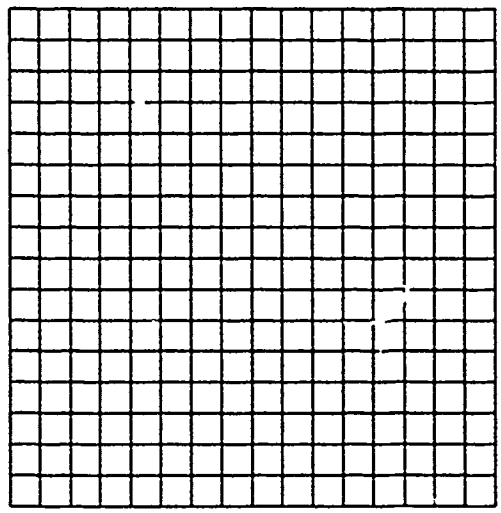


Figure 9. Domain to be decomposed

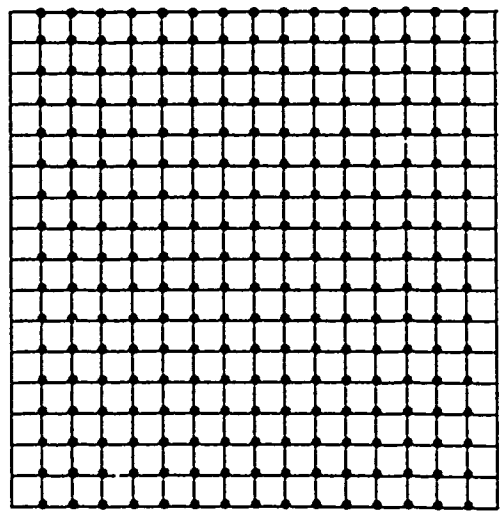


Figure 10. Decomposition D_1—bandwidth minimization

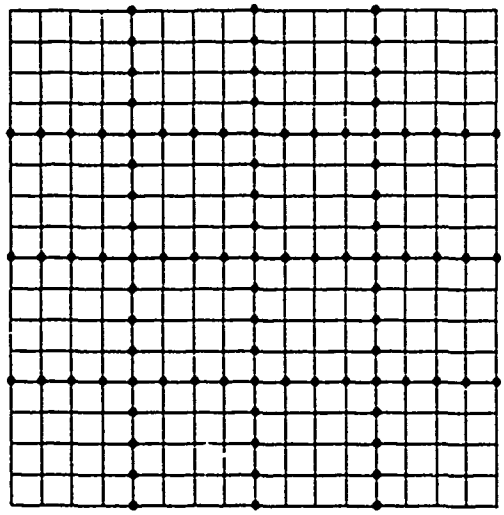


Figure 11. Decomposition D_2—interface minimization

different chips with which any chip needs to communicate. It can be seen (Figure 12) that for D_1 the bandwidth equals 2, while for D_2 it equals 8.

It should be remarked that, if the substructuring approach[14,15] had been chosen—that is assigning a subdomain to a physical processor, D_1 would have been more efficient than D_2. For this decomposition, each chip would buffer the contributions of its interface nodes and send only

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1		X														
2	X		X													
3		X		X												
4			X		X											
5				X		X										
6					X		X									
7						X		X								
8							X		X							
9								X		X						
10									X		X					
11										X		X				
12											X		X			
13												X		X		
14													X		X	
15														X		X
16															X	

Figure 12(a). Interchip communication pattern for D_1

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1		x			x	x										
2	x		x		x	x	x									
3		x		x		x	x	x								
4			x				x	x								
5	x	x				x			x	x						
6	x	x	x		x		x		x	x	x					
7		x	x	x		x		x		x	x	x				
8			x	x			x				x	x				
9					x	x				x			x	x		
10					x	x	x		x		x		x	x	x	
11						x	x	x		x		x		x	x	x
12							x	x			x				x	x
13									x	x				x		
14									x	x	x		x		x	
15										x	x	x		x		x
16											x	x			x	

Figure 12(b). Interchip communication pattern for D_2

two messages, one to the chip at its left and another to the chip at its right. The decomposition D_2 requires the same chip to send up to 8 buffered messages. These messages would eventually be shorter, but would still render D_2 more expensive because of message start-up costs. However, we have opted for a virtual processor approach—that is assigning one element to a virtual processor, for reasons that are given in Section 8. For this case, processors exchange information one node at a time, so that the number of interface nodes associated with a decomposition is more important than its bandwidth. The reader can confirm that decomposition D_1 delivers 255 interface nodes, while D_2 delivers only 93. Indeed, there is another equally, if not more important, reason why D_2 is better for the CM-2 than D_1. In the case of D_1, all of the 16 processors of any chip communicate simultaneously with a set of processors which are on the same neighbouring chip (Figure 12). This generates a significant amount of contention for the single wire that connects these two chips. In the case of D_2 however, one can observe (Figure 14) that:

- for each chip, only 12 out of the 16 processors communicate with processors onto another chip
- only 3 processors out of these 12 communicate simultaneously with the same neighbouring chip, so that much less contention occurs for the wire connecting the two chips. We recall that each chip is connected with up to 12 other ones using 12 different wires which can operate in parallel.

The decomposition D_1 was obtained using a general purpose finite element decomposer presented by the first author in Reference 17. We advocate its use in conjugation with the mapper given in Reference 16 for massively parallel computations on the CM-2. The efficiency improvement potential of this preprocessing phase is demonstrated with the following finite element wave propagation problem. Plate 2 shows the discretization of a tapered cantilever beam. The beam is

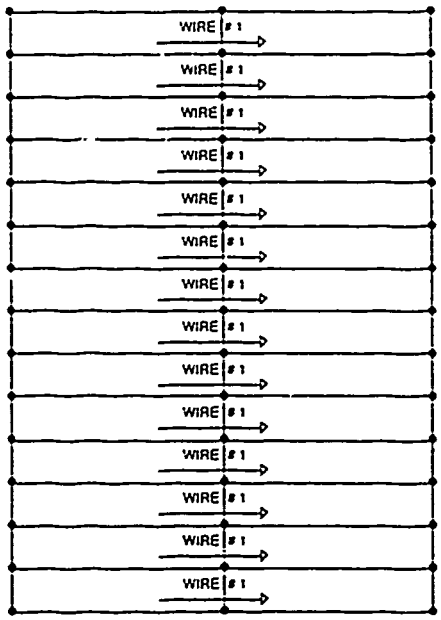


Figure 13. Wire contention induced by decomposition D_1

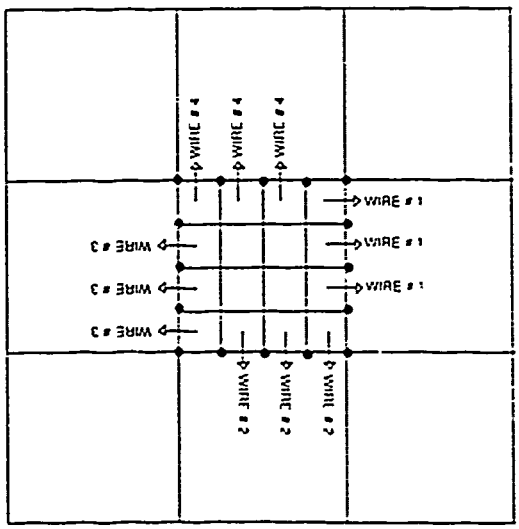


Figure 14. Wire traffic for decomposition D_2

modelled with 4-node isoparametric elements and linearly elstic plane stress constitutive equations. It is fixed at one end and subjected at the tip of the other to an impact point loading. The wave propagation nature of the problem dictates the meshing technique to create elements which are, as far as possible, of equal size. Since the beam is tapered, transition zones with irregular elements had to be introduced. Other mesh irregularities are due to the presence of a region with

a hole. The complete mesh contains 8192 elements, which corresponds to an 8K CM_2. The use of a naive mapping (element i into processor $i-1$) would have resulted in a maximum routing distance between adjacent elements equal to 9. Our decomposer/mapper reduces this distance to 5. If EFF denotes the efficiency (speed-up per processor) of the parallel computations using a naive mapping, and f is the factor by which the decomposer/mapper reduces the maximum routing distance between adjacent elements, the theoretical improved efficiency[16] is given by

$$\mathrm{EFF}^* = \frac{1}{\left(1 - \frac{1}{f}\right) + \frac{1}{fE}} \tag{5}$$

For this problem, we have measured an efficiency EFF = 40 per cent on an 8K CM_2. Since $f = 9/5$, the predicted improved efficiency is EFF* = 54 per cent. A second run of the problem using the decomposer/mapper has revealed a measured improved efficiency EFF* = 60 per cent. The discrepancy between the predicted and measured improved efficiencies is due to the fact that (5) does not account for the wire contention problem.

6. FLOWCHART OF THE MASSIVELY PARALLEL TRANSIENT SIMULATION

The overall organization of the solution on the CM_2 of a transient dynamic problem using the explicit central difference algorithm is depicted in Figure 15. It consists of four phases, namely: mesh preprocessing, data loading, number crunching and data unloading.

A conservative stable time step for the central difference algorithm is given by

$$h \leqslant \frac{2}{\omega_{\max}^{(e)}} \tag{6}$$

where $\omega_{\max}^{(e)}$ is the maximum element frequency of the undamped dynamic problem. Belytschko has pointed out that it is in fact usually not practical to compute the maximum eigenvalues of the element directly, for this would increase the cost of computation considerably.[19] Instead, formulas for upper bounds on $\omega_{\max}^{(e)}$ have been recommended. However, on massively parallel

Read Input File (Front End)
Decompose Mesh and Form Parallel Data Structure (Front End)

Load Parallel Data Structure (Front End - CM_2)

Compute Lumped Mass Matrix (CM_2)
Compute Critical Time Step (CM_2)
Loop on Time Steps (Front End)
{
Compute Internal and External Local Forces (CM_2)
Assemble Global Forces (Interprocessor Communication)
Compute Velocities, Displacements, Strains and Stresses (CM_2)

Visualize Results (CM_2 - Frame Buffer)
Archive Results (CM_2 - Data Vault)
}

Figure 15. Solution of a transient problem on the CM_2

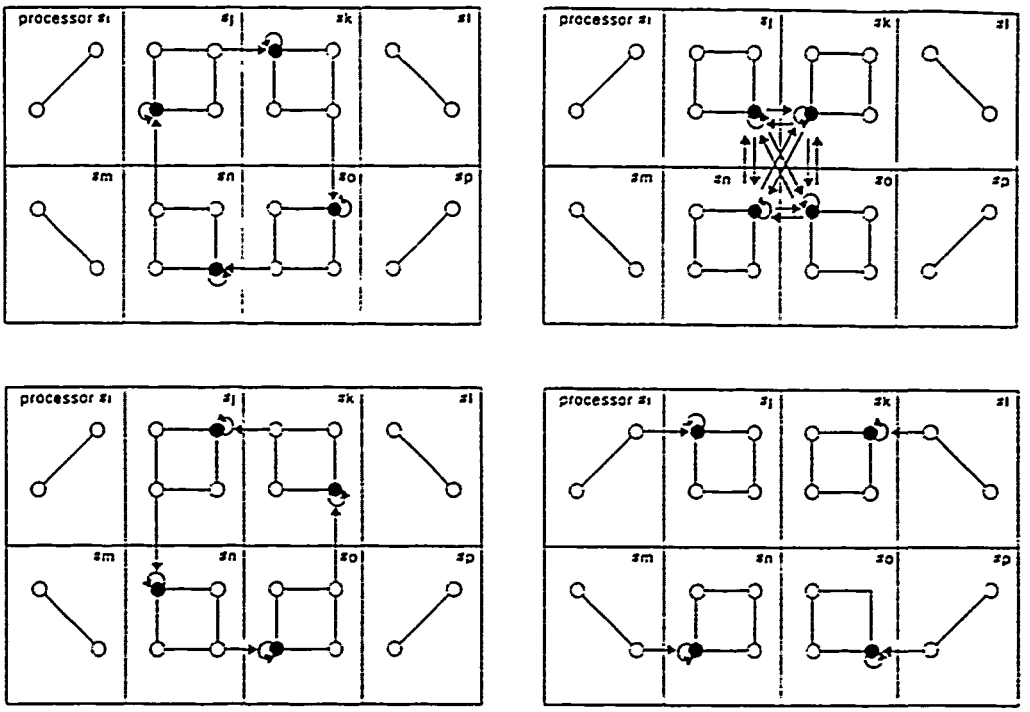


Figure 16. Interprocessor communication for a hybrid patch

processors such as the CM_2, the parallelism inherent in the computation of $\omega_{max}^{(e)}$ is such that this frequency is obtained at the cost of the frequency of one single element.

The interprocessor communication mechanism for a mesh with more than one type of element is illustrated in Figure 16. For the example shown, the 4-node elements are activated first. They communicate in four steps, one node at a time. Next, the 4-node elements are de-activated and the truss elements are selected. These communicate in two steps. As explained in Section 2.2, the serialization between different types of elements is due to the SIMD nature of the CM_2.

7. EXAMPLES

In this section, we apply our approach to massively parallel finite element explicit computations to the solution of various transient problems on an 8K CM_2 with Weitek accelerators. We analyse performance results in detail. We assess the efficiency of our decomposition/mapping strategy at reducing communication time. We highlight the impact on machine performance of variations in mesh topology, finite element modelling and problem non-linearities. We also report on the performance of the Data Vault system for problems that are I/O bound.

For each example, two simulations were carried out. The first one assumed a linear elastic material. In the second simulation, the material was assumed to have an elastoplastic behaviour governed by a von Mises yield condition.

7.1. E1: Transient response of a cracked aluminium plate

The quarter of a mesh in Figure 17 was generated to study the dynamic response of a cracked aluminium plate under a uniform time varying loading. The full mesh contained a total of 4008 plane stress elements and 4073 nodes. Mesh irregularities were induced by transition zones. The *NEWS grid* could not be used.

7.2. E2: Wave propagation in a three-dimensional bar

The second example considered was the impact of a metallic ball on an unsupported glassy bar. The bar was discretized using 8160 brick elements (Figure 18). The finite element mesh contained 13 500 nodes and 40 500 degrees of freedom. Given the regularity of the discretization, the *NEWS grid* was used for interprocessor communication. This example was also re-run using the router for performance comparison.

7.3. E3: Shuttle docking induced vibrations in a space station

This dynamic analysis was carried out to investigate the vibrations of a space station model assembled from 5-m erectable struts. These vibrations were assumed to be induced by a shuttle docking. The finite element model (Figure 19) comprised 7584 three-dimensional truss elements and 2304 nodes. It was generated by aligning identical cells along various axes. However, each cell by itself was irregular and did not allow the use of the *NEWS grid*.

7.4. E4: Three-dimensional glassy bar on an elastic foundation

The wave propagation example problem E2 was repeated with different boundary conditions. The glassy bar was assumed to be supported by a layer of foam. The mesh was comprised of

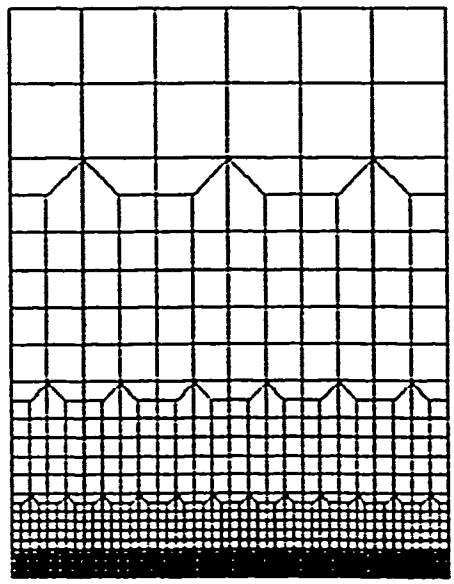


Figure 17. A quarter of a mesh for a cracked plate

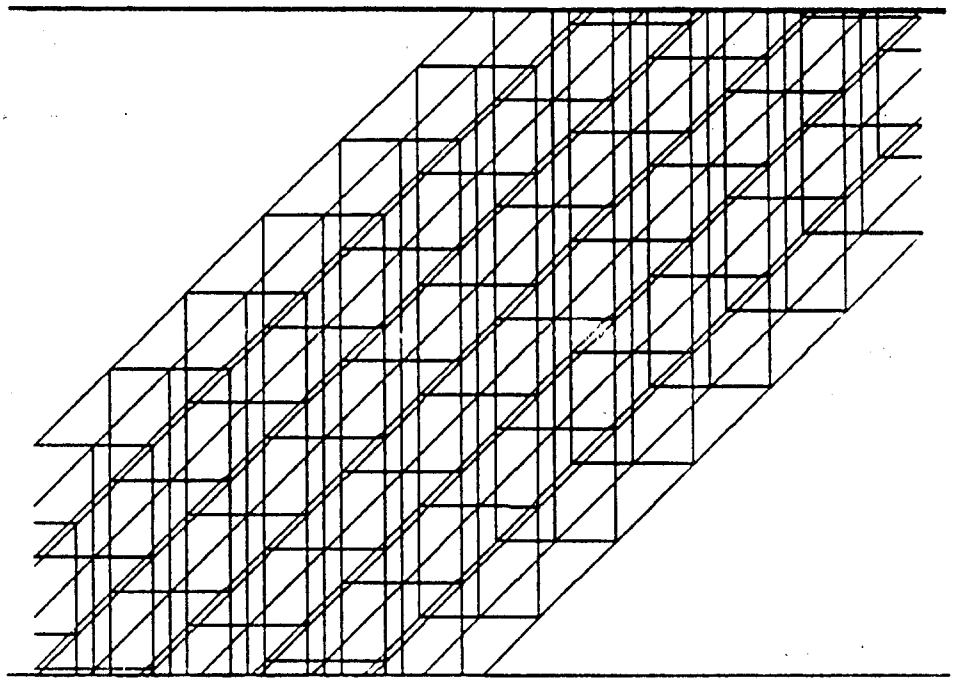


Figure 18. Finite element discretization of a glassy bar

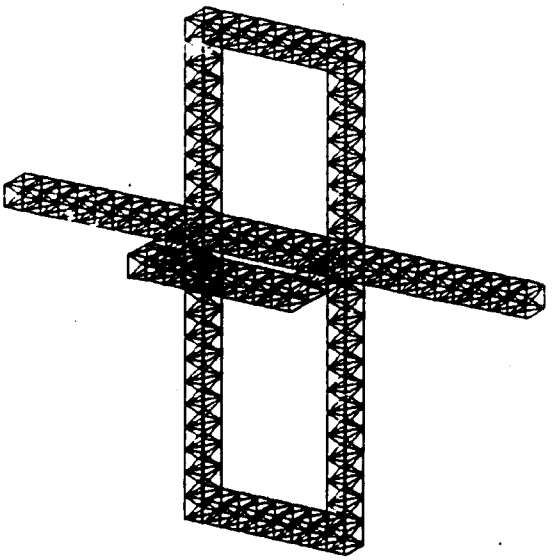


Figure 19. A space station model

a total of 8164 elements (which is very close to the number of elements in the former mesh), of which 1636 truss elements were used to model an elastic foundation.

7.5. *Performance results and analysis*

The large majority the code segments was written in C*. Occasionally we have used Paris functions to speed up some manipulations. Floating-point arithmetic was performed in single precision (32 bit words). Measured performance results are gathered in Tables II. III, IV, V and VI. The reported Mflops rates account for every integer and floating point operation, whether used for addressing or number crunching. Only example E2 could make use of the *NEWS grid*. However, all timings, except those given in Table VI, correspond to runs where communication was carried through the *router*. Execution times are given in seconds and correspond to a sample of 2000 time integration steps and a vp ratio equal to 1.

Table II. Overall measured performance for various transient finite element computations

Example	Mesh pre-processing	Data loading in the CM_2	Equation of motion solving	Sustained Mflops
E1—elastic	1·04 sec	5·47 sec	861 sec	400
E1—elastoplastic	1·04 sec	5·47 sec	1033 sec	480
E2—elastic	1·98 sec	31·78 sec	4139 sec	392
E2—elastoplastic	1·98 sec	31·78 sec	4718 sec	440
E3—elastic	1·28 sec	13·56 sec	887 sec	254
E3—elastoplastic	1·28 sec	13·56 sec	896 sec	256
E4—elastic	2·11 sec	33·00 sec	4770 sec	340
E4—elastoplastic	2·11 sec	33·00 sec	5440 sec	386

Table III. Data Vault system performance

Example	Solving equation of motion	Unloading results on front end	Unloading results on Data Vault
E1	861 sec	5340 sec	3·81 sec
E2	4139 sec	16400 sec	12·61 sec
E3	887 sec	9500 sec	7·04 sec

Table IV. Computation vs. communication

Example	Solving equation of motion	Computation time	Communication time
E1	861 sec	460 sec	401 sec
E2	4139 sec	1959 sec	2180 sec
E3	887 sec	260 sec	627 sec
E4	4770 sec	2340 sec	2430 sec

Table V. True communication time

Example	Computation time	Effective communication time	Software overhead
E1	460 sec	81 sec	320 sec
E2	1959 sec	1380 sec	1280 sec
E3	260 sec	146 sec	481 sec

Table VI. Router vs. *NEWS grid*

Example	Computation time	Communication time using the *NEWS grid*	Communication time using the router
E2	4139 sec	560 sec	2660 sec

The mesh pre-processing phase corresponds to the decomposition of the finite element mesh, as explained in Section 5. It also includes the setup of the finite element parallel data structure, which is then distributed across the processors. Both of these phases are shown to require relatively very little computer time. It can also be observed that, in the worst case, the non-linear computations consume only about 15 per cent additional time. This is due to the explicit nature of the radial return mapping algorithm that was used. Because of 'what you see is what you get', the reported Mflop rates should be compared to those measured in Section 3 and not to the theoretical peak performance of the machine. It should also be noted that our C* code still leaves room for further optimizations.

For examples E1, E2, and E3, the computed displacements, strains and stresses were archived on secondary storage after each time integration step. Two solutions were compared. In the first case, these results were brought back to the front end and stored in appropriate disk files. For that case, the measurements given in Table III demonstrate that the amount of involved I/O dominated the simulation total time. In the second case, the results were transferred in parallel directly to a Data Vault system. The speed-up provided by the Data Vault is shown to be of the order of 1400! This parallel I/O capability is what was most lacking on earlier hypercubes.[18]

If T_{cp} and T_{cm} are respectively the computation parallel time and the communication parallel time, and N_p is the number of available processors on a given parallel machine, the achieved efficiency (speed-up per processor) can be expressed as

$$\mathrm{EFF} = \frac{1}{N_p}\frac{N_p T_{cp}}{T_{cp} + T_{cm}} = \frac{1}{1 + \dfrac{T_{cm}}{T_{cp}}}$$

The results given in Table IV indicate that efficiencies of 53, 47, 29 and 49 per cent are achieved respectively for examples E1, E2, E3 and E4. If one refers to the performance results of Section 3, it can be seen that the sustained Mflop rates reported in Table II are consistent with these efficiencies. At the first glance, these efficiency results appear to be very pessimistic. However, they are well above the 10 per cent often obtained on current vector supercomputers.[20] The reader can observe that the timing results for example E4 are very close to the cumulative timings of examples E2 and E3, which illustrates the impact of the SIMD nature of the CM-2 on the MIMD

nature of finite element computations. It should also be noted that, while the communication time is fixed for a given mesh, the computation time increases with the complexity of the analysis. Thus, highly non-linear formulations which include large deformations are expected to yield higher efficiencies than those deduced from Table IV.

At this point, we give further details regarding interprocessor communication in the context of finite element explicit computations. As outlined in Section 5, the finite elements of a mesh exchange their local contributions one node at a time. For a given finite element, this information exchange procedure is organized around two nested loops. The outer loop is carried over the nodes that are connected to this element. The inner loop is carried over the neighbouring elements that are attached to each local node. Using a C notation, this is written as:

```
for (node = 1; node ⩽ my_nodes; node++)                    (7)
    {
        start = pointer[node]; stop = pointer[node + 1] − 1;
        for (position = start; position ⩽ stop; position++)   (8)
        {
        neighbor = proc_att_to_node [position];
        exchange (variable, myself, neighbor);
        }
    }
```

where *my_nodes* is the total number of nodes that are connected to a given finite element and *proc_att_to_node* is the array containing the identification of the neighbouring elements. Clearly, these variables are element dependent. The total number of communications to be performed by one processor is determined by the product $P_{cm}^{(e)} = d*(pointer[my_nodes + 1] - 1)$ which is both element and mesh dependent. The CM_2 being an SIMD machine, the communication time is determined by $\max_e\{P_{cm}^{(e)}\}$. For a regular mesh composed of three-dimensional truss elements ($d = 3$) or 4-node plane elements ($d = 2$), every node is attached to 4 elements, so that 24 communication instructions per time integration step are required for the truss element and 32 for the 4-node plane element. However, Table IV indicates that the space station example exhibits a longer communication time than does the aluminium plate problem. The reason is that in the mesh of example E3, some truss elements are connected to 12 other elements. Because of the SIMD nature of the CM_2, the element with the highest degree of connectivity determines the communication time. For a regular mesh with 8-node solid elements ($d = 3$) each time integration step is followed by 192 communication steps, since each node can be attached up to eight different elements. This is reflected in Table IV where example E2 is shown to possess by far the longest communication time (2180 sec). In summary, the amount of communication involved in finite element explicit computations on the CM_2 is determined by the element topology and order, and the mesh irregularities. Because only d nodal information is exchanged at a time among the CM_2 processors, three-dimensional and high order elements substantially increase the communication time. Mesh irregularities also adversely affect the amount of communication because of the SIMD nature of the CM_2. It is interesting to note that elements which transmit physical information across edges and faces such as those proposed by De Veubeke,[21] would require much less communication than traditional elements. These elements should be revisited for computations on massively parallel processors such as the CM_2.

An in-depth investigation of the communication phase was carried out. It was found that most of the communication time was elapsed in the header of loop (8). This loop header involves the quantities *start* and *stop* which differ from one processor to another in the presence of mesh irregularities and different element types. Consequently, the front-end computer has to process and manage several different loops rather than a unique one, which is not very efficient on an SIMD machine. The time associated with the headers of loops (7) and (8) is referred to as software overhead in Table V. The true time that is elapsed in effective communication among the processors is shown to be only a fraction of the overall communication time (see Table V).

Because it was designed to handle arbitrary meshes, our C* code did not make use of the *NEWS grid* package. However, a special module that incorporated calls to the *NEWS grid* was written specifically for the regular mesh of example E2. Execution times for this example using both the *NEWS grid* and the *router* are shown in Table VI. Clearly, a high price is paid for the handling of eventual mesh irregularities.

However, the irregular pattern of communication is fixed in time. Thus, a considerable improvement can be achieved if this pattern is evaluated at the first time step, then somehow stored in the CM_2 for use during subsequent time steps. We believe that this is an issue that massively parallel computer architects should investigate.

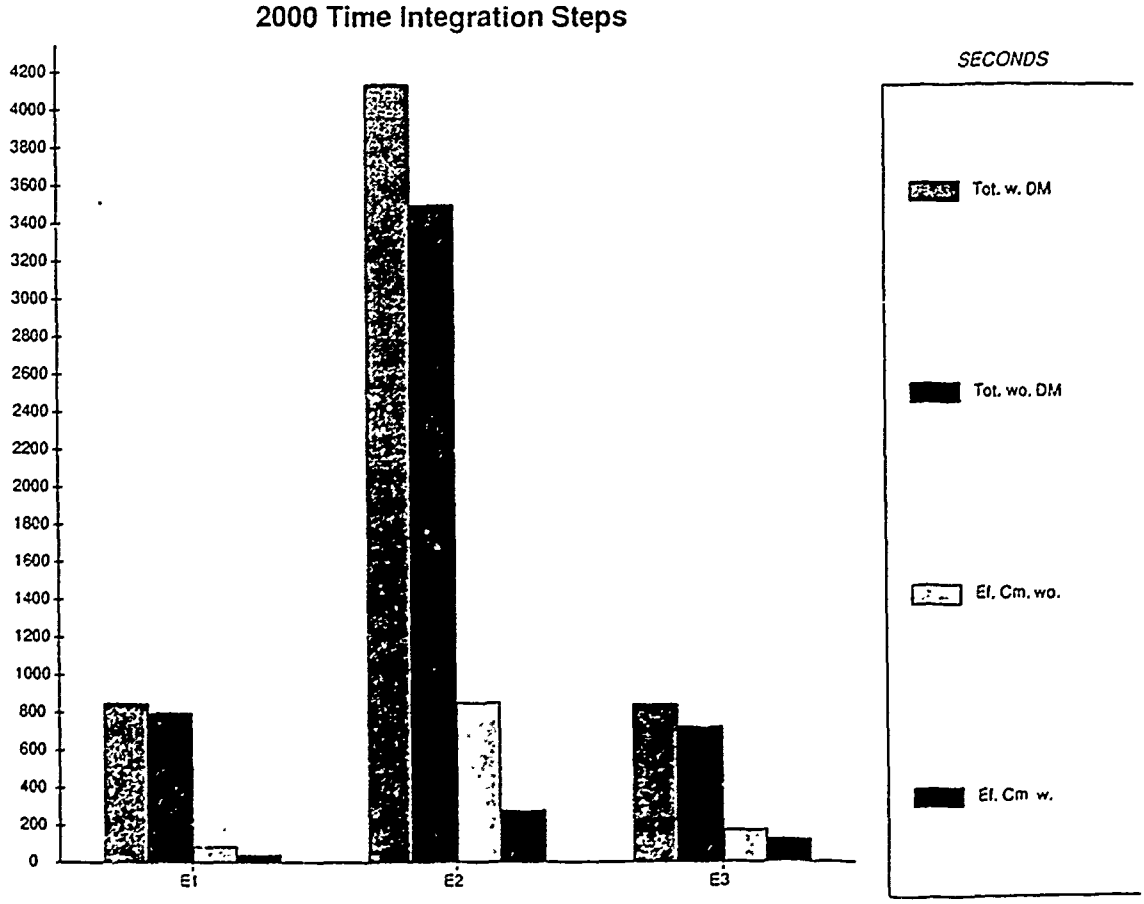


Figure 20. The decomposer/mapper performance

In order to assess the performance of the decomposer/mapper module, examples, E1, E2 and E3 were re-run with the naive shifted identity mapping (element i in processor $i-1$). Figure 20 demonstrates that the true communication time can be reduced by as much as 60 per cent. Unfortunately, the total execution time is reduced only between 10 and 17 per cent because of the communication software overhead associated with mesh irregularities.

8. CONCLUDING REMARKS

We have reported herein on our experience in performing transient finite element computations on the CM_2. We have presented the architectural features of this parallel processor and discussed their impact on finite element computational strategies. In particular, those features which distinguish the CM_2 from earlier hypercubes have been emphasized. These include the virtual processor concept and the fast parallel I/O capabilities. The processor memory memory size of 64 Kbits has been shown to penalize high order elements. We have also described and discussed a domain decomposition strategy and a mapping algorithm which are suitable for massively parallel processors such as the Connection Machine. The main idea behind the decomposition technique is the minimization of both the amount of wire contention within a chip, and the amount of communications between different chips. A given finite element mesh is partitioned into 16-element subdomains which correspond to the 16-processor chips of the Connection Machine. This partitioning is carried out in a way that minimizes the number of nodes at the interface between the subdomains. As a result, only those processors which are mapped onto finite elements at the periphery of a subdomain communicate with processors packaged on different chips. Moreover, this partitioning is such that the connectivity bandwidth of the resulting subdomains is large enough to allow an efficient use of the interchip wires. The mapping algorithm attempts to reduce the distance information has to travel through communication network. In essence, the algorithm searches iteratively for an optimal mapping through a two-step minimization of the communication costs associated with a candidate mapping. Various issues related to the single instruction multiple data stream nature of the CM_2 and pertinent to computational mechanics have been addressed. Measured performance results for realistic two- and three-dimensional transient problems have been reported. Three-dimensional and high order elements have been shown to induce longer communication times. Mesh irregularities have been shown to slow down the computation speed in many ways. The Data Vault has been demonstrated to be very effective at reducing the I/O time.

Now, we briefly highlight some additional implementational and theoretical issues that we hope will materially advance the application ranges of finite element computations on this highly parallel processor.

8.1. Virtual processor ratio vs. substructuring

In this work, we have assigned when possible more than one finite element to a single processor using the virtual processor feature of the CM_2. However, another way to obtain the same result is to assign a substructure to an individual processor.[14,15] From a numerical point of view, both approaches are equivalent. However, these two distinct approaches differ in their implementations and may perform differently. The substructure approach requires each processor to work with both external and internal data structures. The set of external data structures stores information about substructure interconnections. These are similar to the ones described in this paper. The set of internal data structures stores the connectivity table of the elements within

a substructure. The computations within each substructure are carried out by looping over the elements of that substructure. The advantage of this approach is a saving in storage since the substructure internal nodes are uniquely defined, and a faster computation of the results associated with these nodes. Moreover, the global results at the internal nodes can be accumulated without any explicit call to a message-passing function. The global quantities at the boundary nodes are accumulated using the router and the external data structures. However, the substructuring approach requires that the sequencer broadcast the same instruction several times, once for each element of the substructure, which increases the overall wall clock execution time. Moreover, this approach does not allow the Weitek chip to pipeline the computations over the elements of the substructure.

On the other hand, the virtual processor approach requires that each element communicate explicitly with its neighbours, even if these are assigned to the same processor. Of course, this communication is virtual since it is within the processor itself and generates minimal additional overhead. On the positive side, the virtual processor approach utilizes only one type of data structure and exploits the pipelining capabilitieis of the Weitek chip. The latter feature significantly enhances overal performance, as demonstrated in Section 3. Consequently, we advocate the use of the virtual processor ratio rather than the substructuring technique, especially if the processor memory size is to be increased in the future.

8.2. Implicit algorithms and the CM_2

In this report, no attempt has been made to design a novel parallel algorithm for the solution of the differential equation of motion. We have selected the central difference algorithm because of its inherent parallelism, which allowed us to focus on implementational issues and to fully explore the multiprocessing capabilities of the CM_2. Our experience suggests that a whole class of explicit and semi-implicit dynamic and static algorithms can be implemented on the CM_2 in a very similar way. Among others, we cite the EBE algorithms,[22] the EBE preconditioners[23] and the Jacobi preconditioned conjugate gradient algorithm.[24] However, the solution of some static and transient problems may necessitate the use of an implicit algorithm, which usually implies the solution of a set of simultaneous banded equations. If the global symmetric stiffness matrix $\mathbf{K}$ is banded, with semi-bandwidth b, then it is well known (see for example Ortega and Voigt[25]) that Gaussian elimination methods for solving $\mathbf{Kd} = \mathbf{F}$ allow at each step on the order of $b^2/2$ pairs of $(+, x)$ to be processed concurrently, but require significant communication because the b entries of the pivot column must be made available to all other processors. Several parallel algorithms based on these elimination methods were designed for finite element applications and were implementated on earlier hypercubes (see for example, Farhat and Wilson[26] and Utku *et al.*[27]). Typically, a processor was assigned to a set of matrix columns. Results from our previous experience with the early version of Intel's iPSC suggests that direct solvers are feasible on hypercubes only when the number of available processors, N_p, is much smaller than the bandwidth b of the given finite element problem, so that communications do not dominate computations. On the iPSC-1, a message that was sent from one extreme corner of a 5-dimensional cube to the other would result in an elapsed time 475 times longer than the time to perform a floating point multiplication (see Rudell.[28]). However, on a 10-dimensional subcube of the CM_2 we have measured the ratio of a broadcast to a floating point computation to be only about 2.87. This observation suggests that, for problems with $b > 360$, a processor could be mapped onto a few matrix entries and a parallel direct solver could be feasible on the CM_2. For problems with smaller bandwidth, direct solvers which operate on more than one pivot at a time[29,30] should also be investigated for implementation on massively parallel processors.

There is an additional issue which has to be examined before attempting to solve finite element equations on the CM_2 with a parallel direct solver. This issue is related to the balance on massively parallel processors between the number of available processors, N_p, and the processor memory size. Let M^n denote a two-dimensional regular n by n finite element mesh, where n is the number of elements along one side. If d is the number of degrees of freedom at a given node, the semi-bandwidth of M^n is $b = d(n + 3)$ and the total number of mathematical unknowns is $N = d(n + 1)^2$. For this mesh, the storage cost of K amounts to $Nb = d^2(n + 3)(n + 1)^2$ words. The total amount of storage available on the CM_2 is $S = N_p * m_p$, where N_p is the number of available processors and $m_p = 8$ Kbytes is the current size of the processor memory. Let $NE = n^2_{max}$ be the maximum number of elements for which M^n has a banded stiffness matrix that can be factored in-core on the CM_2. Table VII gives the values of NE for different values of d and for the case of a fully configured Connection Machine ($N_p = 65\,536$). Values of NE are shown for both single precision (32 bit words) and double precision (64 bit words) floating point arithmetic.

Clearly, except for the case where $d = 2$ and floating point arithmetic is done in single precision, NE is smaller than N_p. Similarly, the case where M^n is an n by n three-dimensional regular mesh is assessed in Table VIII for various values of d.

For this case, NE is much smaller than N_p, even for $d = 2$ and for single precision floating point arithmetic. For $d = 6$ (some shell elements), only 8000 elements (4000 elements) can be included in M^n when computations are carried out using single precision (double precision) floating point arithmetic.

It is noted that the eventual solution of a system of equations is only one phase of several finite element computational sequences. In linear three-dimensional analysis, this phase dominates the computer execution time. However, in the non-linear analysis of flexible space structures most of the computational time is usually spent in modules that perform element level computations.[31] These include the evaluation of generalized nodal internal forces and/or elemental stiffness matrices. Consider now a mesh M^n where the number of elements NE is chosen so that the upper part of the banded stiffness matrix K fills the N_p processor memories completely. The preceding complexity analysis demonstrates that the balance on the CM_2 between the number of processors and the memory size of each processor is such that NE is much smaller than N_p. Hence, if a direct algorithm is used to solve a finite element system of equations, the N_p processors will be active during the solution phase, but $N_p - E$ processors will remain idle during the rest of

Table VII. Number of allowable elements vs. DOF/node for the two-dimensional case

$N_p = 65\,536$		$d = 2$	$d = 3$	$d = 4$	$d = 5$	$d = 6$
Single precision	NE	102400	59536	40401	29929	23409
Double precision	NE	64009	37249	25281	18769	14884

Table VIII. Number of allowable elements vs. DOF/node for the three-dimensional case

$N_p = 65\,536$		$d = 2$	$d = 3$	$d = 4$	$d = 5$	$d = 6$
Single precision	NE	29791	19683	13824	10648	8000
Double precision	NE	19683	12167	9261	6859	4913

the phases which involve element level computations. Consequently, an in-core direct solution strategy would not efficiently utilize the computational power of the CM_2 in a highly non-linear finite element analysis.

ACKNOWLEDGEMENTS

The authors wish to acknowledge the support by the Naval Research Laboratory (NRL) under Grant DOD N00014-87K-2018, with Dr Hank Dardy and Dr Louise Schuetz as technical monitors. The first author also acknowledges the support by the National Science Foundation under grant ASC-8717773. Both CM_2s at NRL and at CAPP were utilized to develop this work. The parallel I/O experiments were done using the Data Vault system at NRL. Special thanks to Bob Whaley (Thinking Machines Corporation and NRL), Eric Hoffman (NRL) and Roldan Pozo (CAPP).

REFERENCES

1. C. Farhat and E. Wilson, 'A new finite element concurrent computer program architecture', *Int. j. numer. methods eng.*, 24, 1771–1792 (1987).
2. G. A. Lyzenga, A. Raefsky and B. H. Hager, 'Finite elements and the method of conjugate gradient on concurrent processors', *CalTech/JPL Report C3P-119*, California Institute of Technology, Pasadena, CA, 1984.
3. T. Belytschko and N. Gilbersten, 'Concurrent and vectorized mixed time, explicit nonlinear structural dynamics algorithms', in A. K. Noor (ed.), *Parallel Computations and Their Impact on Mechanics*, American Society of Mechanical Engineers, New York, 1987, pp. 279–290.
4. C. Farhat and L. Crivelli, 'A general approach to nonlinear FE computations on shared memory multiprocessors', *Comp. Methods Appl. Mech. Engng*, 72, 153–172 (1989).
5. M. Benten, C. Farhat and H. Jordan, 'The force for efficient multitasking on the CRAY series of supermultiprocessors', *Proceedings of the Fourth International Symposium on Science and Engineering on CRAY Supercomputers*, Minneapolis, Minnesota, Oct. 12–14, 1988, pp. 389–406.
6. D. W. White and J. F. Abel, 'Bibliography on finite elements and supercomputing', *Commun. appl. numer. methods*, 4, 279–294 (1988)
7. A. K. Noor, 'Parallel processing in finite element structural analysis', in A. K. Noor (ed.), *Parallel Computations and Their Impact on Mechanics*, American Society of Mechanical Engineers, New York, 1987, pp. 253–277.
8. C. Farhat, 'Parallel computational strategies for large space and aerospace flexible structures. Algorithms, implementations and performance', *Proceedings, Supercomputing in Engineering Structures*, Oberlech, Austria, IBM Europe Institute, July 11–15, 1988.
9. O. McBryan, 'New architectures: Performance highlights and new algorithms', *Parallel Comp.*, 7, 477–499 (1988).
10. J. L. Gustafson, G. R. Montry and R. E. Benner, 'Development of parallel methods for a 1024-processor hypercube', *SIAM J. Sci. Statist. Comp.*, 9, 609–638 (1988).
11. R. D. Krieg, 'Unconditional stability in numerical integration methods', *J. Appl. Mech. ASME*, 40, 417–521 (1973).
12. W. Daniel Hillis, *The Connection Machine*, MIT Press, Cambridge, Mass, 1987.
13. Thinking Machines Corporation, 'Connection Machine model CM_2 technical summary', *Technical Report Series*, HA87-4, 1986.
14. C. Farhat, E. Wilson and G. Powell, 'Solution of finite element systems on concurrent processing computers', *Eng. Comp.*, 2, 157–165 (1987).
15. Fox *et al.*, *Solving Problems on Concurrent Processors*, Prentice-Hall, Englewood Cliffs, N.J., 1988.
16. C. Farhat, On the mapping of massively parallel processors onto finite element graphs', *Comp. Struct.*, 32, 347–354 (1989).
17. C. Farhat, 'A simple and efficient automatic FEM domain decomposer', *Comp. Struct.*, 28, 579–602 (1988).
18. J G. Malone, 'Automated mesh decomposition and concurrent finite element analysis for hypercube multiprocessors computers', *Comp. Methods Appl. Mech. Eng.*, 70, 27–58 (1988).
19. T. Belytschko, 'Overview of semidiscretization', in T. Belytschko and T. J. R. Hughes (eds.), *Computational Methods for Transient Analysis*, North-Holland, 1983, pp. 1–63.
20. Kuck *et al.*, 'The effects of program restructuring, algorithm change, and architecture choice on program performance', *IEEE*, C-27, 129–138 (1984).
21. M. Geradin (ed.), *B. M. Fraejis De Veubeke Memorial Volume of Selected Papers*, Sijthoff & Noordhoff, 1980.
22. T J. R. Hughes *et al.*, 'Element-by element implicit algorithms for heat conduction', *J. Eng. Mech.*, 109, 576–585 (1983).
23. T J. R Hughes, R M. Ferencz and J. O. Hallquist, 'Large-scale vectorized implicit calculations in solid mechanics on a Cray X-MP/48 utilizing EBE preconditioned conjugate gradients', *Comp. Methods Appl. Mech. Eng.*, 61, 215–248 (1987).

24. G. H. Gloub and C. F. Van Loan, *Matrix Computations*, The John Hopkins University Press, 1983.
25. J. M. Ortega and R. G. Voigt, 'Solution of partial differential equations on vector and parallel computers', *SIAM Rev.*, 27, 149–240 (1985).
26. C. Farhat and E. Wilson, 'A parallel active column equation solver', *Comp. Struct.*, 28, 289–304 (1988).
27. S. Utku, M. Salama and R. Melosh, 'Concurrent factorization of positive definite banded Hermitian matrices', *Int. j. numer. methods eng.*, 23, 2137–2152 (1986).
28. R. Rudell, 'Parallel processing efficiency on the hypercube, *CS252 Project Report*, The University of California at Berkeley, 1985.
29. G. Alaghband and H. F. Jordan, 'Multiprocessor sparse L/U decomposition with controlled fill-in', *ICASE Report No. 85–48*, NASA Langley Research Center, Hampton, Virginia 23665, 1985.
30. F. J. Peters, 'Parallel pivoting algorithms for sparse symmetric matrices', *Parallel Comp.*, 1, 99–110 (1984).
31. N. F. Knight *et al.*, 'CSM testbed development and large scale structural applications', *Proceedings of the 4th International Symposium*, Minneapolis, Minnesota, 1988, pp. 359–388.

International Journal of High Speed Computing, Vol. 1, No. 2 (1989) 289-302
© World Scientific Publishing Company

DYNAMIC FINITE ELEMENT SIMULATIONS ON THE CONNECTION MACHINE

C. FARHAT, N. SOBH and K. C. PARK
Department of Aerospace Engineering Sciences
Center for Space Structures and Controls
and Center for Applied Parallel Processing
University of Colorado at Boulder
Boulder, CO 80309-0429

Received November 15, 1988
Revised February 17, 1989

ABSTRACT

This paper reports on our early experience with solving large scale finite element dynamic problems on the Connection Machine. We describe a distributed data structure that allows very efficient massively parallel computations on irregular two and three dimensional finite element meshes. The often encountered mesh irregularities inhibit the use of the NEWS communication package. We propose an alternative approach to interprocessor communication that is based on an optimal mapping of the processors onto the finite elements of the irregular mesh. Our parallel data structure and processor mapping are applied to finite element wave propagation problems on the CM-2.

Keywords: massively parallel, explicit computations, finite element.

1. Introduction . Here we are interested in solving large-scale structural problems on the Connection Machine. The differential equations of motion arising from a finite element (FE in the sequel) formulation [1] are expressed as:

$$M\ddot{u} + D\dot{u} + Ku = f \tag{1}$$

where

$$(2) \qquad M = \sum_{e=1}^{e=E} m^{(e)} \;;\; K = \sum_{e=1}^{e=E} k^{(e)} \;;\; D = \sum_{e=1}^{e=E} d^{(e)} \;;\; f = \sum_{e=1}^{e=E} f^{(e)};$$

and

$$(3) \qquad \begin{aligned} m_{ij}^{(e)} &= \int_{\Omega^{(e)}} \rho H_i H_j \, d\Omega \\ k_{ij}^{(e)} &= \int_{\Omega^{(e)}} [LH_i]^T C [LH_j] \, d\Omega \\ d_{ij}^{(e)} &= \alpha m_{ij}^{(e)} + \beta k_{ij}^{(e)} \\ f_i^{(e)} &= \int_{\Omega^{(e)}} f H_i \, d\Omega \end{aligned}$$

In equation (1) above, $\ddot{u}$, $\dot{u}$ and u denote respectively the second, first and zero derivatives of the vector of nodal displacement unknowns u_i. The summations in equation (3) should be interpreted as boolean sums. They define the assembly process of the mass matrix M, the damping matrix D, the stiffness matrix K and the load vector f, from the corresponding element level submatrices m, d, k and f. In equation (3), H, L and C are the matrix representations of, respectively, an interpolating operator, a spatial derivative operator and a given constitutive equation.

It is important to note that implicit algorithms for the solution of (1) usually require the assembly of the global matrices M, D, K and f. This is because they usually involve the factorization of a linear combination of these matrices. However, most explicit algorithms work directly at the element level, and operate on the submatrices m, d, k and f.

Much work has been devoted to the solution of static and dynamic finite element equations on both shared memory and local memory multiprocessors. Extensive list of references can be found in references [2], [3] and [4], among others. However, very few results have been reported for massively parallel processors such as the Connection Machine. In this paper, we focus on implementational aspects of massively parallel finite element computations and report on our early experience with the CM-2. The remainder of this paper is organized as follows. Section 2 addresses storage issues which are more severe than those usually encountered on hypercubes with fewer processors. Section 3 describes the selected data structure. In section 4, an optimal processor mapping is introduced. The parallel data structure and the optimal processor mapping are applied in Section 5 to finite element wave propagation problems. Section 6 offers some concluding remarks.

2. Storage Considerations for the CM-2. In this paper we consider only the case of a lumped mass and a Coulomb damping. In other words, both the mass matrix, M, and the damping matrix, $D = \alpha M$, are diagonal. We denote by $f_{ex}(t)$ the vector of external time dependent prescribed nodal forces f, and by $f_{in}(u)$ the vector of internal stiffness nodal forces Ku, so that equation (1) takes the following form:

$$(4) \qquad M\ddot{u} + D\dot{u} + f_{in}(u) - f_{ex}(t) = 0$$

Equation (4) above is integrated with the fixed-step central difference algorithm:

$$(5) \qquad \begin{aligned} \dot{u}^{n+1/2} &= \dot{u}^{n-1/2} + hM^{-1}(f_{ex}(t^n) - f_{in}(u^n) - D\dot{u}^n) \\ u^{n+1} &= u^n + h\dot{u}^{n+1/2} \end{aligned}$$

where u is the nodal displacement vector, h is the fixed time step, and the superscript n indicates the value at the discrete time t^n. Note that the diagonal form of the global mass matrix M preserves the explicit nature of the central difference time integration algorithm.

At each time step t^n, equation (5) can be solved using only element-level computations. The global mass matrix M and damping matrix D are evaluated by assembling only once the element-level submatrices $m^{(e)}$ and $d^{(e)}$. The equalities

$$f_{in}(u^n) = Ku^n = \sum_{e=1}^{e=E} k^{(e)} u^{n(e)} = \sum_{e=1}^{e=E} f_{in}^{(e)}(u^{n(e)})$$

$$f_{ex}(t^n) = \sum_{e=1}^{e=E} f_{ex}^{(e)}(t^n)$$

show that, at each time step t^n, $f_{in}(u^n)$ and $f_{ex}(t^n)$ can also be directly assembled from, respectively, the element-level internal forces $f_{in}^{(e)}(u^{n(e)})$ and

the element-level external forces $f_{ex}^{(e)}(t^n)$. Hence, if each processor of the CM-2 is mapped onto one or several finite elements, equation (4) can be efficiently integrated in parallel. First, all processors concurrently compute their local contributions $(D\dot{u}^n)^{(e)}$, $f_{ex}^{(e)}(t^n)$ and $f_{in}^{(e)}(u^{n(e)})$. Next, these contributions are efficiently accumulated by requiring that each processor which is mapped onto an element e communicate its local contributions $f_{in}^{(e)}(u^{n(e)})$, $f_{ex}^{(e)}(t^n)$ to $f_{in}(u^n)$ and $f_{ex}(t^n)$, only to those processors which are mapped onto elements that are adjacent to e. This should allow the message-passing on a hypercube to be parallelized, since each processor communicates only with a very few other processors.

Clearly, the above parallel scenario assumes that each processor stores in its local memory the element data involved in each element-level computation. In particular, it implies that each processor stores the elemental stiffness matrix $k^{(e)}$, and the diagonal mass matrix $m^{(e)}$. The following example shows that this can be problematic.

Consider the 9-node curved shell element shown in Figure 1. To each node are attributed three displacements and two rotations, which amount to a total of 45 degrees of freedom per element. Consequently, the symmetric part of the elemental stiffness matrix contains $45*(45+1)/2 = 1035$ words. If double precision is used, the storage of $k^{(e)}$ amounts to $1035*64 = 66240$ bits, which exceeds the 65536 bits that are available on a single CM-2 processor. On the other hand, if single precision is used, the storage of $k^{(e)}$ requires 33120 bits so that 32416 bits are left for the storage of the vectors $u^{(e)}$, $\dot{u}^{(e)}$, $m^{(e)}$, $d^{(e)}$, $f_{ex}^{(e)}$, and $f_{in}^{(e)}$. However, even in the latter case, only a virtual processor ratio of 1 can be used, which limits the size of the finite element mesh to the maximum number of processors available on the Connection Machine at hand.

It is well known that because at time step t^{n+1}, K is involved in the solution process only in the product $f_{in}(u^n) = Ku^n$, the above storage requirements can be considerably decreased by bypassing the evaluation of $k^{(e)}$ and directly computing $f_{in}(u^n)$. This can be done by replacing in the finite element formulation the quantity $\int_{\Omega(e)} |LH_i|^T C |LH_j| \, d\Omega$ with its equivalent $\int_{\Omega(e)} |LH_i|^T \sigma \, d\Omega$, where at t^{n+1}, σ is evaluated explicitly from the displacements computed at t^n. As a result, the solution process defined in (5) involves only vector quantities which do not require a large amount of storage, so that fairly high virtual processor ratios are possible.

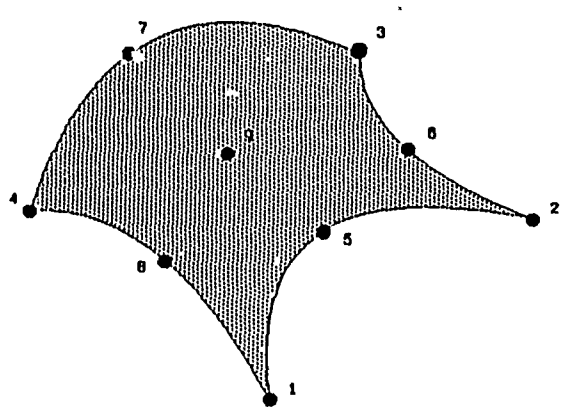


Fig. 1. A 9-node shell element

3. A Distributed Data Structure. Explicit time integration algorithms are best parallelized at the element level. Each processor performs a substantial amount of local computations ($\ddot{u}^{n(e)}$, $\dot{u}^{n(e)}$, $u^{n(e)}$, $f_{ex}^{(e)}(t^n)$, $f_{in}^{(e)}(u^{n(e)})$) before briefly communicating to a few neighboring processors its contributions $f_{in}^{(e)}(u^{n(e)})$ and $f_{ex}^{(e)}(t^n)$ to the global internal and external forces at a given node, $f_{in}(u^n)$ and $f_{ex}(t^n)$. In this section, we describe the finite element data structures with which the parallel computational strategy can best be implemented on the CM-2. These can be grouped into two sets. The first set of data structures deals with element-level parallel computations. The second set provides the mechanism for parallel interprocessor communications.

For the sake of simplicity, assume that each processor of the CM-2 is mapped onto a single element of a given mesh.

To be able to perform locally its assigned element-level computations - that is, to perform these computations without interacting with the front-end machine -, each processor must store in its own memory its element type (truss, beam, shell, ..., number of Gauss points, ...), its element material properties (density, parameters and coefficient for constitutive equation, damping characteristics, thickness, ...), its nodal geometry (nodal coordinates, number of nodes per element), and its boundary conditions (fixed/free degrees of freedom at each node, prescribed forces at each node). This infor-

mation is compacted in one-dimensional arrays. In addition, each processor must also store in its memory a set of scalars corresponding to computational parameters such as the fixed time step h, and a scalar or one-dimensional buffer for the temporary storage of messages to be passed to neighboring processors.

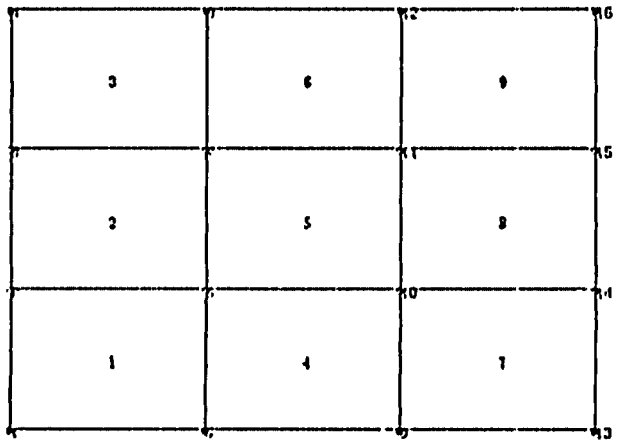


Fig. 2. Regular finite element mesh

Consider the regular finite element mesh shown in Figure 2. Except on the boundaries, each element is connected in the same pattern to exactly eight other elements. Consequently, each processor communicates with its neighbors in the same manner. Interprocessor communication can be performed through two sweeps of the NEWS mechanism. These two sweeps are depicted in Figure 3.

However, the beauty of the finite element method resides in the fact that it solves models with irregular meshes. Typically, a finite element mesh consists of several patches which are connected together using irregular transition regions (Fig. 4).

Hence, in the general and practical case, in order to be able to communicate $f_{in}^{(e)}(u^{n(e)})$ and $f_{ex}^{(e)}(t^n)$, each processor must be able to identify at run time:

- the set of adjacent elements (processors)
- the node(s) it shares with each adjacent element (processor)

- for each common node, the correct sequence of degrees of freedom which are common to adjacent elements. This is necessary to guarantee that, for example, a moment is not accumulated with a force, or that a force in the x direction, is not accumulated with a force in the y direction.

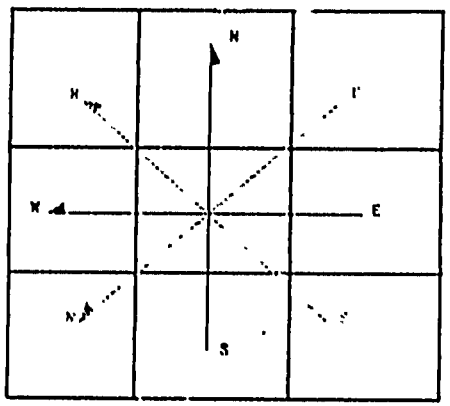


Fig. 3. Two NEWS sweeps

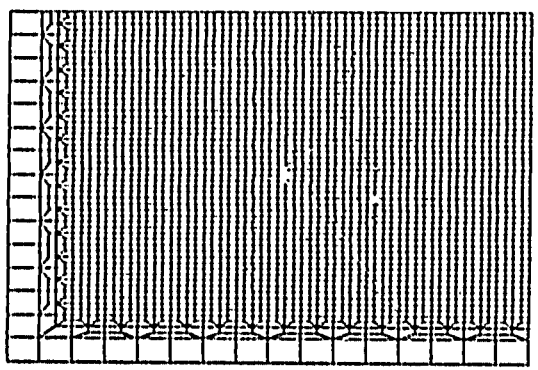


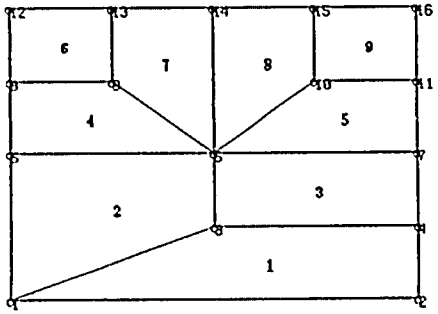
Fig. 4. Typical finite element mesh

If the above information is gathered in a global form on the front-end machine, most of the execution time elapsed during the accumulation phase would be due to message-passing between the CM_2 processors and the front-end computer. On the other hand, if this information is decentralized - that is, if each processor memory is loaded with only the piece of that information which is relevant to the connectivity of the element that is assigned to the processor-, the accumulation phase can be performed without any message-passing between the CM_2 and the front-end computer. Consequently, prior to any dynamics computation, we load each processor memory with the following one-dimensional arrays:

PRATNO This is the restriction of the node connectivity array to the element that is assigned to a given processor. It contains for each node of the assigned element, the list of adjacent elements (processors) which are connected to this node. These are stored in a stacked fashion.

PPRATN This is a pointer array. It stores in position i, the location in PRATNO of the list of elements (processors) attached to the node in the $i-th$ local position.

LOCATT For each entry in PRATNO, this array specifies the local position of the common node in the adjacent element (processor).



Element 1
PRATNO [2,3,3,2]
PPRATN [1,2,2,3,5]
LOCATT [1,2,1,2]

Fig. 5. A distributed data structure for interprocessor communication

The above arrays are set up by a dedicated finite element mesh analyzer which was presented in reference [5]. Their average storage cost is about 80 integer words per processor. Clearly, this is a very small overhead. Figure 5 illustrates their mechanism.

4. Mapping the CM_2 onto Irregular FE Meshes. Since the mesh irregularities inhibit the exploitation of the NEWS mechanism, we rely on the data structures of Section 3 to guide the router during interprocessor communication. However, there is still one additional problem to resolve. Efficiency in massively parallel computations requires the minimization both of the distance that information must be communicated and of the "hammering" on the router. In the case of finite element computations, this implies that adjacent elements must be assigned, as much as possible, to directly connected processors. This defines the mapping problem - that is, it defines which hardware processor is to be mapped onto which finite element of a given mesh.

A heuristic algorithm for mapping massively parallel processors onto finite element graphs has been presented in reference [5]. Basically, the algorithm searches iteratively for a better mapping candidate through a two-step minimization procedure of the communication costs associated with a specific parallel processor topology. Because it seeks a very fast solution for a machine with thousands of processors, this algorithm does not guarantee "the" optimal mapping. However, it has produced very encouraging results on a variety of non-uniform two and three-dimensional meshes. As an example, consider the finite element discretization of an aircraft flexible structure which is shown in Figure 6. For the sake of clarity only 1024 elements have been drawn; however, the mesh comprises 65536 shell elements.

The 65536 processors of the CM_2 are packaged into 4096 16-processor chips, each having its own router node. The 4096 router nodes are arranged in a hypercube of dimension 12. To cope with this topology, we first apply our mesh decomposer [6] and constrain each submesh to contain exactly 16 adjacent finite elements. The result is a decomposition of the given mesh into 4096 submeshes, each containing 16 connected finite elements. Next, we apply our heuristic mapper to identify which hardware chip is to be mapped onto which submesh. Finally, within each submesh, the elements are numbered randomly between the chip number and the chip number + 15. For the finite element mesh above, our mapper results in a maximum distance between communicating processors which is equal to 4. Mapping

processor i onto element $i+1$ would have given a maximum distance of 12 between processors, which is three times larger.

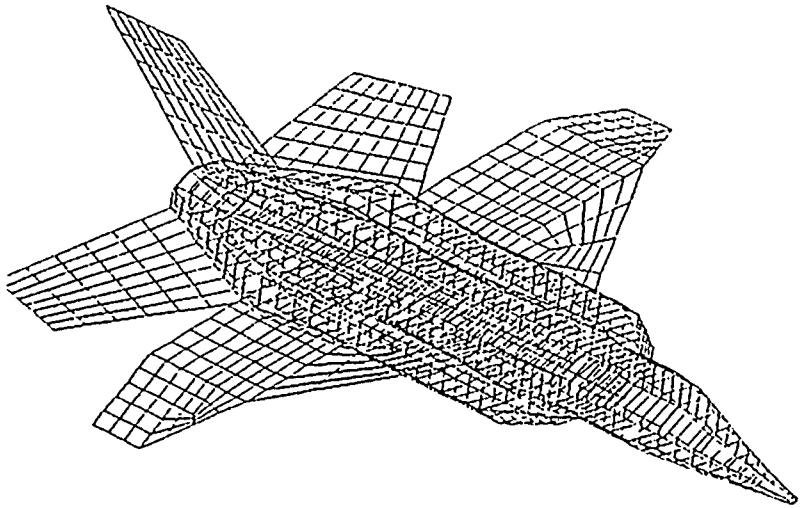


Fig. 6. Finite element discretization of an aircraft

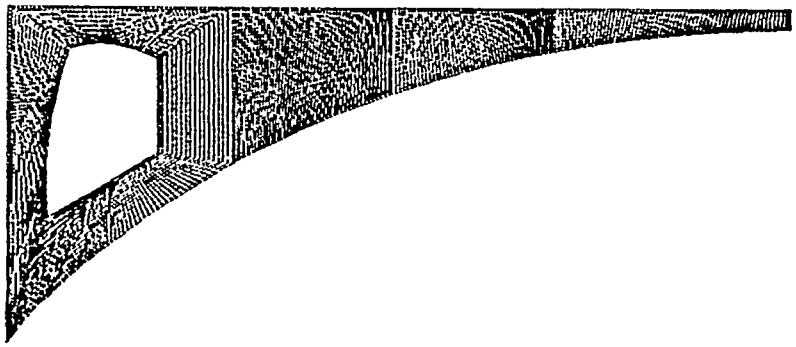


Fig. 7. FE discretization of a tapered beam

5. Finite Element Wave Propagation Problems on the CM-2.

Problem Description

Figure 7 shows the finite element discretization of a tapered cantilever beam. The beam is modeled with 4-node isoparametric elements and linearly elastic plane stress constitutive equations. It is fixed at one end and subjected at the tip of the other to an impact point loading.

The wave propagation nature of the problem dictates the meshing technique to create elements which are as far as possible, of equal size. Since the beam is tapered, transition zones with irregular elements had to be introduced (Fig. 8). Other mesh irregularities are due to the presence of a region with a hole.

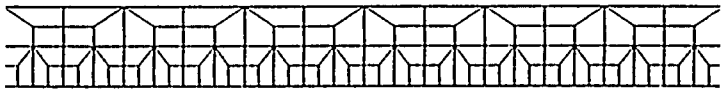


Fig. 8. Irregular transition zone

The complete mesh contains 8192 elements, which corresponds to an 8K Connection Machine. The mapper of reference [5] is used to assign the finite elements to the processors. The maximum distance between adjacent elements is 5. The use of a trivial mapping (element i into processor $i-1$) would have resulted in a maximum distance between adjacent elements equal to 9.

*Implementation on an 8K CM-2 using C**

An 8K subcube of a 16K Connection Machine located at Naval Research Laboratory, Washington, is used to assess the performance of our approach to massively parallel finite element dynamic computations. The major part of the coding is written in C^*, which is a parallel version of C. The code segments which deal with parallel I/O are written in Paris (Parallel Instruction Set). These are used, for example, to load initially the data in the processors, and to bring the final results back to the front-end computer, at some specified time steps. Floating point arithmetic is done with 32 bit words.

Measured Performance

At this writing, C^* does not support some of the floating point reduction capabilities and does not exploit the hardware features for indirect address-

ing. Consequently, C^* is slower than the other languages which are available for the CM_2. The performance results for the problem above are reported below. The measurements correspond to 10000 integration time steps.

Loading the data
sequentially: 9.22 seconds
in parallel : 0.76 second

Parallel Computations
computations: 5000 seconds
communications: 992 seconds

Bringing results to front-end
sequentially: 15000 seconds
in parallel : 1460 seconds

In the above, "sequentially" refers to fetching the processors one after another within a loop. Buffering the finite element data and loading it in parallel into the processors using Paris system calls speeds up this operation by a factor greater than 10 over the sequential mode.

The parallel computation phase consumes 5992 seconds, of which 992 seconds elapse in interprocessor communication. The reader should note that for a given element the amount of communication is fixed, and is usually proportional to three factors: the number of nodes that are attached to this element, the number of degrees of freedom per node, and the number of adjacent elements. However, for a given element, the amount of local computations depends on the complexity of this element (interpolation order, linear/nonlinear, ...). In other words, complex elements such as fully nonlinear shell elements have a better computation/communication ratio and consequently will attain a better performance than the 4-node linear plane stress element used here.

At each time step, the computed displacements, velocities and accelerations are brought back to the front-end computer for storage on a disk. Ultimately, this data will be stored on the CM_2 data vault system which provides parallel I/O facilities, but which is not yet available at NRL. The timings for the parallel and serial transmissions of the results to the front-end show a speed-up factor of 10, which is illustrative of the potential of the data vault system where parallelism is exploited not only during transmitting the data, but also during writing it on separate disks. I/O manipulations are extremely important in finite element dynamic computations; the Connection Machine has a great potential for speeding them up.

A floating point operation counter was inserted in the code. The CM_2 sustained a 0.560 gigaflop performance rate during the finite element dynamic computations. This number is linearly scaled for a 64K machine.

6. Conclusions. An early experience with solving finite element dynamic problems on the CM_2 has been reported. The problem of handling irregular grids has also been addressed. A solution consisting of guiding the router with a set of distributed data structures and minimizing the time elapsed in interprocessor communication by using an optimal mapping of the processors onto the finite element mesh has been implemented and tested.

Irregular meshes and irregular patterns of discretization (presence of different types of element within the same mesh) are best handled with an MIMD style of programming. This may seem unrealistic for an SIMD machine such as the CM_2. However, the C^* language offers several ways to mimic an MIMD execution of the processes on the Connection Machine. For example, each element of an irregular mesh can be surrounded by a different amount of adjacent elements. Consequently, each processor may have to communicate with a different number of other processors. Using C^*, each element is declared as a "domain". Furthermore, the communication phase is coded within a loop where the index is declared to be "poly" (parallel or private) and ranges between 1 and the number of surrounding processors. Unfortunately, this mechanism introduces an extra overhead. Since the instructions are broadcast from the front-end computer to the CM_2 processors, a loop with a "poly" index executes several times slower than a loop with a "mono" index, that is the same index for all the processors. This is because in the former case, the front-end has to manage more than one loop, while in the latter case, the same loop index is broadcast at any time to all the processors. Consequently, for efficient programming, not only the interprocessor communications must be minimal, but also the broadcast of the instructions from the front-end computer to the computational processors must be optimized.

The timings reported in section V for the finite element dynamic computations demonstrate an efficiency rate of effective utilization of the processors equal to $(1/(1 + 992/5000) = 83\%$. Since rates of performance as high as 3.00 gigaflops are possible on the CM_2 [7], this means that one can expect a

performance as high as 2.5 gigaflops for our finite element massively parallel computations. The discrepancy between this expected performance and the measured sustained rate of 0.560 gigaflop is due to the current inaptitude of the C^* compiler at generating optimal code. As soon as C^* becomes as efficient as $*Lisp$, one can expect our code to reach the 2.5 gigaflop rate of performance on a 64K Connection Machine.

Acknowledgments. The authors wish to acknowledge the support by the Naval Research Laboratory under Grant DOD N00014-87K-2018, with Dr. Hank Dardy and Dr. Louise Schuetz as technical monitors. They also would like to thank Bob Whaley of Thinking Machines Corporation at NRL for his valuable contributions to the development of the present finite element code, and John Krystynak and Eric Hoffman at NRL for making their graphics utilities on the frame buffer available for us.

REFERENCES

[1] T. J. R. HUGHES, *The Finite Element Method*, Prentice Hall, N. J., 1987.

[2] FOX et. al, *Solving Problems on Concurrent Processors*, Prentice Hall, N. J., 1988.

[3] D. W. WHITE AND J. F. ABEL, *Bibliography on finite elements and supercomputing*, Communications in Applied Numerical Methods, 4, No. 2, (1988), pp. 279-294.

[4] A. K. NOOR, *Parallel processing in finite element structural analysis*, in Parallel Computations and Their Impact on Mechanics, A. K. Noor, ed., American Society of Mechanical Engineers, New York, 1987, pp. 253-277.

[5] C. FARHAT, *On the mapping of massively parallel processors onto finite element graphs*, Computers & Structures, 32, No 2, (1989), pp. 347-354.

[6] C. FARHAT, *A simple and efficient automatic FEM domain decomposer*, Computers & Structures, 28, No. 5, (1988) pp, 579-602.

[7] O. MC BRYAN, *State-of-the-art in highly parallel computer systems*, in Parallel Computations and Their Impact on Mechanics, A. K. Noor, ed., American Society of Mechanical Engineers, New York, 1987 pp. 31-48.

THE ANS SHELL ELEMENTS: EARLIER DEVELOPMENTS AND RECENT IMPROVEMENTS

K. C. Park and E. Pramono
Department of Aerospace Engineering Sciences
and Center for Space Structures and Controls
University of Colorado
Boulder, Colorado

G. M. Stanley and H. A. Cabiness
Computational Structural Mechanics Section
Lockheed Palo Alto Research Laboratory
Palo Alto, California

ABSTRACT

Thin shell elements whose formulation is based on the assumed natural-coordinate strain(ANS) fields are presented. The present shell element construction offers several improvements over the earlier versions of the ANS elements (Ref. 28), particularly regarding the inplane shear, twist and transverse shear strains because of changes in the shell normals. These improvements were made possible by adopting the new formulation offered in Ref. 25 and by introducing new ways of interpolating assumed strain fields. The elements thus constructed correctly preserve rigid motions, exhibiting no locking for skewed element shapes such as hemispherical geometries. The new 9-ANS element resulting from the present construction possesses a significantly improved modeling of transverse shear strains, which may be important for composite analysis.

1. INTRODUCTION

This paper describes the construction of the shell elements based on assumed natural-coordinate strains (ANS) that began with the linear interpolation of membrane strains (Ref. 24), and the 9-ANS shell element (Refs. 28-30 and 38-40). In doing so, we rely on several important corrections from the theoretical formulation in Ref. 25. The corrections were motivated by our desire to base our element construction on the formulation that incorporates as much shell behavior as possible from the outset into the variational equations of motion to be discretized, to improve the curvature effects, and to directly incorporate into the assumed strain-displacement relations a thick-shell capability allowing transverse shear deformations (Ref. 35). This paper may, therefore, be regarded as our earnest effort to effect a "marriage a la mode" between the finite element method and shell theories.

The impetus for developing the previous ANS shell elements was to improve the element performance when the elements become progressively distorted. This was, in essence, accomplished by abandoning the standard isoparametric mapping used to transform the natural-coordinate derivatives into their inertially fixed Cartesian counterparts. In addition, concepts

such as the Hrennikoff grid (Ref. 9), a series of consistent interpolations on the natural-coordinate strain terms (Ref. 21), a tensorial transformation of the natural coordinate strains into the corresponding Cartesian strains, and directionally selective, reduced integration were blended to avoid element locking and spurious mechanisms. The resulting elements, thus, acquired one important theoretical property: the strains remain invariant for an arbitrary choice of the local coordinate system, hence, improving the element performance for distorted grids.

Subsequent numerical evaluations of the ANS elements indicated that the 9-noded ANS element (or 9-ANS element) manifested no ostensible deficiency for production level applications. The 9-ANS element was used to analyze a series of complex shell problems, such as the post-buckling problem of a curved composite panel with a cutout (Ref. 38-40), and the pear-shaped shell and cylinder with cut outs (Ref. 18), which increased our confidence in the element.

On the more rigorous theoretical aspects of the 9-ANS and 4-ANS elements, there emerged two hard evidences. First, the 4-ANS element almost locks the solution for a pinched hemispherical problem unless the reduced one-point integration is invoked instead of the full four-point integration. Second, the 9-ANS element, when put to the patch test, exhibited an oscillation on the constant strain state with an amplitude of about one-tenth of a percent of the constant strains. While the first pathology is avoidable and the second is harmless in practice, these two pathologies motivated us to reconstruct the ANS elements to eradicate such isolated pathologies as reported in Ref. 37, thus, making the ANS shell elements free from "exceptions" in performance. We will refer to the original ANS element construction as "the old construction" and to the present ANS element construction as the "the new construction."

In the old construction of the ANS shell elements, the Hrennikoff lattice lines were chosen. In the 4-ANS element, the four grid edges were chosen to be the Hrennikoff lines. In the 9-ANS element, the four edge lines and the two natural-coordinate lines, $\xi = 0$ and $\eta = 0$, were chosen to be the Hrennikoff lines. Then, the derivatives of the covariant displacements along the Hrennikoff lines were interpolated which were termed as the covariant physical strains along the Hrennikoff lines. The natural-coordinate strains in the element interior were then obtained by interpolating the appropriate covariant displacement derivatives along the Hrennikoff lines according to the isoparametric interpolation weights. The orthogonal shell-coordinate or inertial-coordinate strains at any point in the element were finally obtained by tensorial transformation of the natural-coordinate strains. This meant two consequences. First, there must be a congruency between the directions of the natural-coordinate strains and those of the natural-coordinate basis vectors. Second, we had to abandon the isoparametric transformation of the natural-coordinate derivatives into the Cartesian ones as first introduced in Ref. 13.

A closer examination of the interpolated natural-coordinate strains showed the above congruency requirements are met only "in the large," even though the incongruency level quickly diminished as the grids were refined. A pathology of the 4-ANS element manifested for skewed grids because the errors committed in interpolating the normal vectors become significant enough to lock the solution. To mitigate this incongruency, we chose the following as the basis for the new ANS element construction.

The most fundamental aspect of the new formulation in Ref. 25 is in the choice of its coordinate system: an inertially fixed coordinate system for translational motions, an orthogonal shell-surface coordinate system for rotational motions, and a natural-coordinate system for strains. The proper use of these three coordinate systems led to the mitigation of several element deficiencies, heretofore present in our earlier versions of the ANS shell elements.

The incremental strain-displacement relations we will employ are expressed on the deformed shell geometries. However, they remain valid for finite-strain and finite-rotation increments and, hence, they can be used both for linear and nonlinear analyses. In particular, the present element construction, based on the strain increments can be easily interfaced with an element-independent corotational procedure (Ref. 34) to effect an efficient nonlinear analysis procedure.

In the construction of the new ANS shell elements, we preserve the two essential ingredients in the old ones: the natural-coordinate strains and the tensorial transformation of the natural-coordinate strains into any desirable orthogonal components. However, in constructing the natural-coordinate strains, we abandoned the old way of interpolating the covariant displacements that vary their directions along the natural-coordinate lines. Instead, we chose to interpolate the inertial Cartesian displacement components to obtain the natural-coordinate covariant strains, because the displacements expressed in the inertial coordinate system do not vary their directions along the natural-coordinate lines. In other words, the covariant strains are expressed directly in the inertial displacement components and their derivatives along the natural-coordinate lines. Finally, the locking-free and mechanism-free measures adopted in the old construction are carried over almost intact into the new construction. We now describe the new construction of ANS-shell elements in full detail.

2. THEORETICAL PRELIMINARIES FOR THIN SHELLS

We summarize the equations of motion for thin shells derived in Ref. 25 and the associated strain-displacement relations.

2.1 Kinematics and Shell Geometries

The position vector of the particle point P (Fig. 1) on the deformed shell is given by

$$\mathbf{r} = \mathbf{r}^* + \zeta \mathbf{b}_3 \tag{1}$$

where

$$\mathbf{r}^* = x\mathbf{e}_1 + y\mathbf{e}_2 + z\mathbf{e}_3, \quad (x \;\; y \;\; z) = \{(X + u) \;\; (Y + v) \;\; (Z + w)\} \tag{2}$$

in which x, y, and z are the *deformed* neutral shell surface position coordinates u, v and w are the displacements measured in the inertial e-system, ζ is the distance of the material point P from the shell neutral surface measured in the b-system attached on the deformed cross-section of the shell, and the vector b is related to the vector e by

$$\mathbf{b} = \mathbf{R}\mathbf{e} \tag{3}$$

The angular velocity of a particle point, P, on the shell cross section is thus given as

$$\tilde{\omega} = -\dot{\mathbf{R}}\mathbf{R}^T = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{4}$$

The variational pseudo-vector, $\delta\alpha$, conjugate with ω, is given by

$$\delta\tilde{\alpha}^T = \delta\mathbf{R}\mathbf{R}^T, \qquad \omega = [\omega_1 \;\; \omega_2 \;\; \omega_3]^T, \qquad \delta\alpha = [\delta\alpha_1 \;\; \delta\alpha_2 \;\; \delta\alpha_3]^T \tag{5}$$

The variational displacement vector, $\delta\mathbf{r}$, is obtained from Eq. 1 as:

$$\delta\mathbf{r} = \delta\mathbf{u}^T\mathbf{e} + \zeta\delta\tilde{\mathbf{u}} \tag{6}$$

where the pseudo-rotation vector, $\delta\tilde{\mathbf{u}}$, is related to the shell-surface, pseudo-rotation quantities, $\delta\boldsymbol{\beta}$ according to

$$\delta\tilde{\mathbf{u}} = [-\mathbf{t}_2^T(2) \;\; \mathbf{t}_2^T(1)] \begin{Bmatrix} \delta\beta_1 \\ \delta\beta_2 \end{Bmatrix}, \quad \delta\tilde{\boldsymbol{\beta}} = \mathbf{T}_t^T\delta\tilde{\mathbf{u}}\mathbf{T}_t, \tag{7}$$

which $t^T_{\ell\beta}(i)$ represents the i-th row of the transformation matrix, $T_{\ell\beta}$, defined by

$$s = T_{\ell\beta}\, e \tag{8}$$

s are attached to the deformed shell surface; and, $T_{\beta s}$ relates the shell-surface basis vectors o the inertial basis vectors according to

$$b = T_{\beta s}\, s \tag{9}$$

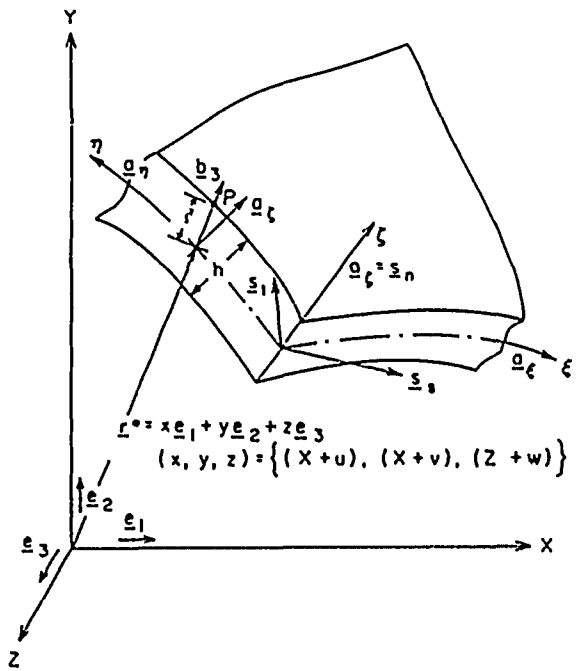


Fig. 1 - Deformed Shell Geometrics in Inertial (e), Shell Surface (s), Natural-Coordinate (a) and Deformed Normal (b_3) Vectors.

The covariant natural-coordinate unit vectors are then obtained by:

$$a_\xi = \frac{1}{\lambda_\xi}\frac{\partial r^o}{\partial \xi} = \frac{1}{\lambda_\xi}(x_{,\xi}\, e_1 + y_{,\xi}\, e_2 + z_\xi e_3) = t^T_\xi e \tag{10}$$

$$a_\eta = \frac{1}{\lambda_\eta}\frac{\partial r^o}{\partial \eta} = \frac{1}{\lambda_\eta}(x_{,\eta}\, e_1 + y_{,\eta}\, e_2 + z_\eta e_3) = t^T_\eta e \tag{11}$$

$$a_\zeta = \frac{a_\xi \times a_\eta}{|a_\xi \times a_\eta|} = (x_{,\zeta}\, e_1 + y_{,\zeta}\, e_2 + z_{,\zeta}\, e_3) = t^T_\zeta e \tag{12}$$

where the two fundamental shell surface quantities, λ_ξ and λ_η, are given by:

$$\lambda^2_\xi = \frac{\partial r^o}{\partial \xi}\cdot\frac{\partial r^o}{\partial \xi}, \qquad \lambda^2_\eta = \frac{\partial r^o}{\partial \eta}\cdot\frac{\partial r^o}{\partial \eta} \tag{13}$$

For subsequent applications, we express the above relation in a compact form:

$$a = T_{n\beta}\, e = \begin{Bmatrix} t^T_\xi \\ t^T_\eta \\ t^T_\zeta \end{Bmatrix} e \tag{14}$$

Finally, the covariant partial derivatives are given by:

$$\frac{\partial}{\partial s_\xi} = \frac{1}{\lambda_\xi}\frac{\partial}{\partial \xi} \qquad \frac{\partial}{\partial s_\eta} = \frac{1}{\lambda_\eta}\frac{\partial}{\partial \eta} \qquad \frac{\partial}{\partial s_\zeta} = \frac{1}{\lambda_\zeta}\frac{\partial}{\partial \zeta} \qquad \lambda_\zeta = \frac{1}{2}h(\xi, \eta) \tag{15}$$

where $h(\xi, \eta)$ is the shell thickness.

2.2 Variational Equations of Motion for Thin Shells

The variational equations of motion for shell structures can be expressed as (Refs. 25 or 38):

$$\delta\mathcal{T}^I + \delta\mathcal{T}^S = \delta\mathcal{T}^T + \delta\mathcal{T}^E \tag{16}$$

In which $\delta\mathcal{T}^I$, $\delta\mathcal{T}^S$, $\delta\mathcal{T}^T$ and $\delta\mathcal{T}^E$ are referred to as the "inertia force", the "stiffness force", the "traction boundary force", and the "external force operators", respectively, given by:

$$\delta\mathcal{T}^I = \int_V \rho(\delta u^T \ddot{u} + \delta\alpha^T \tilde{\ell}\ell^T\dot{\omega} + \delta\alpha^T\tilde{\ell}\tilde{\omega}\ell^T\omega)dV \tag{17}$$

$$\delta\mathcal{T}^S = \int_V (\delta\epsilon_{\xi\xi}\sigma_{\xi\xi} + \delta\epsilon_{\xi\eta}\sigma_{\xi\eta} + \delta\epsilon_{\eta\eta}\sigma_{\eta\eta} + \delta\epsilon_{\xi\zeta}\sigma_{\xi\zeta} + \delta\epsilon_{\eta\zeta}\sigma_{\eta\zeta})dV \tag{18}$$

$$\delta\mathcal{T}^T = \int_S (\delta u^T \Gamma^T + \delta\alpha^T\tilde{\ell}S)\begin{Bmatrix}\sigma_1 \\ \sigma_2 \\ \sigma_3\end{Bmatrix} dS \tag{19}$$

$$\delta\mathcal{T}^E = \int (\delta u^T f_e + \delta u^T R^T f_a + \delta q^T R^T f_e + \delta q^T f_a + \delta\alpha^T\tilde{\ell}R f_e + \delta\alpha^T\tilde{\ell}f_a)dV \tag{20}$$

$$f = f^T_e e + f^T_a a \tag{21}$$

where $\tilde{\ell}$ is a (3×3)-skew symmetric matrix formed by $\ell = [0 \quad 0 \quad \xi]$, σ_{ij} and ϵ_{ij} are the conjugate pairs of the natural-coordinate stresses and strains, σ_1, σ_2, and σ_3 are the surface tractions along the ξ, η, ζ-coordinates around the shell element boundaries, and f_e is the inertially applied load and f_a is the follow on force, respectively.

2.3 Variational Natural-Coordinate Strains

The variational natural-coordinate strains specialized for thin shells from Ref. 25 can be summarized as:

$$\begin{Bmatrix} \delta\epsilon_{\zeta\zeta} \\ \delta\epsilon_{\zeta\eta} \\ \delta\epsilon_{\eta\eta} \\ \delta\epsilon_{\zeta\zeta} \\ \delta\epsilon_{\eta\zeta} \end{Bmatrix} = \begin{Bmatrix} \delta\epsilon^{\circ}_{\zeta\zeta} + \zeta\delta\kappa_{\zeta\zeta} \\ \delta\epsilon^{\circ}_{\zeta\eta} + \zeta\delta\kappa_{\zeta\eta} \\ \delta\epsilon^{\circ}_{\eta\eta} + \zeta\delta\kappa_{\eta\eta} \\ \delta\gamma_{\zeta} \\ \delta\gamma_{\eta} \end{Bmatrix} \tag{22}$$

where

$$\delta\epsilon^{\circ}_{\zeta\zeta} = t^T_{\zeta}\frac{\partial\delta u}{\partial S_{\zeta}} \tag{23}$$

$$\delta\epsilon^{\circ}_{\zeta\eta} = t^T_{\zeta}\frac{\partial\delta u}{\partial S_{\eta}} + t^T_{\eta}\frac{\partial\delta u}{\partial S_{\zeta}} \tag{24}$$

$$\delta\epsilon^{\circ}_{\eta\eta} = t^T_{\eta}\frac{\partial\delta u}{\partial S_{\eta}} \tag{25}$$

$$\delta\gamma_{\zeta} = t^T_{\zeta}\frac{\partial\delta u}{\partial S_{\zeta}} + t^T_{\zeta}\delta\hat{u} \tag{26}$$

$$\delta\gamma_{\eta} = t^T_{\zeta}\frac{\partial\delta u}{\partial S_{\eta}} + t^T_{\eta}\delta\hat{u} \tag{27}$$

$$\delta\kappa_{\zeta\zeta} = t^T_{\zeta}\frac{\partial\delta\hat{u}}{\partial S_{\zeta}} + \left[\frac{\partial t^T_{\zeta}}{\partial S_{\zeta}} - t^T_{\zeta}\frac{\partial t_{\zeta}}{\partial S_{\zeta}}\cdot t^T_{\zeta}\right]\frac{\partial\delta u}{\partial S_{\zeta}}^{\dagger} \tag{28}$$

$$\delta\kappa_{\zeta\eta} = \left(t^T_{\zeta}\frac{\partial\delta\hat{u}}{\partial S_{\eta}} + t^T_{\eta}\frac{\partial\delta\hat{u}}{\partial S_{\zeta}}\right)$$

$$+\left(\frac{\partial t^T_{\zeta}}{\partial S_{\zeta}} - \left(t^T_{\zeta}\frac{\partial t_{\zeta}}{\partial S_{\zeta}} + t^T_{\eta}\frac{\partial t_{\zeta}}{\partial S_{\eta}}\right)\cdot t^T_{\zeta}\right)\frac{\partial\delta u}{\partial S_{\eta}}^{\S} \tag{29}$$

$$+\left(\frac{\partial t^T_{\zeta}}{\partial S_{\eta}} - \left(t^T_{\zeta}\frac{\partial t_{\zeta}}{\partial S_{\zeta}} + t^T_{\eta}\frac{\partial t_{\zeta}}{\partial S_{\eta}}\right)\cdot t^T_{\eta}\right)\frac{\partial\delta u}{\partial S_{\eta}}^{\S} \tag{30}$$

$$\delta\kappa_{\eta\eta} = t^T_{\eta}\frac{\partial\delta\hat{u}}{\partial S_{\eta}} + \left[\frac{\partial t^T_{\zeta}}{\partial S_{\eta}} - t^T_{\zeta}\frac{\partial t_{\zeta}}{\partial S_{\eta}}\cdot t^T_{\eta}\right]\frac{\partial\delta u}{\partial S_{\eta}}^{\dagger} \tag{31}$$

in which

$$\frac{\partial u}{\partial S_{\zeta}} = \frac{1}{A_{\zeta}}\left(\frac{\partial u}{\partial\xi} \quad \frac{\partial v}{\partial\xi} \quad \frac{\partial w}{\partial\xi}\right)^T \tag{32}$$

$$\hat{u} = (\hat{u} \quad \hat{v} \quad \hat{w})^T \tag{33}$$

$$\delta\hat{u} = \hat{X}^T_{s\theta}\begin{Bmatrix} \delta\beta_{\eta} \\ -\delta\beta_{\zeta} \\ 0 \end{Bmatrix} \tag{34}$$

It should be noted that the incremental strain-displacement relations are obtained by replacing δ simply by Δ in the preceding equations. The terms designated by $(\,)^{\dagger}$ and $(\,)^{\ddagger}$ in the bending strains of $\delta\kappa_{\zeta\zeta}$ and $\delta\kappa_{\eta\eta}$ usually remain small for thin shells, and they may be neglected in most applications. However, it is recommended that the term designated by $(\,)^{\S}$ in the twist term, $\delta\kappa_{\zeta\eta}$, be retained to satisfy rigid-body motions as discussed in a classical shell formulation (Ref. 36).

2.4 Shell-Coordinate Strains and Stress Increments

The natural-coordinate system, n, is related to the shell-coordinate system, s, from Eqs. 8 and 9 by

$$n = T_{n\theta}\cdot T^T_{s\theta}s = T_{ns}s, \qquad \text{or} \quad s = T_{sn}n \tag{35}$$

where s is defined as

$$s = (s_{\xi} \; s_t \; s_n), \qquad s_n = n_{\zeta} \tag{36}$$

so that not only (s_{ξ}, s_t) are chosen to form an orthogonal system that is tangent to the shell surface but also the normal shell-surface vector, s_n, coincides with the natural-coordinate vector, n_{ζ}.

The variational stiffness force operator (Eq. 18) in the shell-surface coordinate system-i.e., s-system-can be written as:

$$\delta\mathcal{F}^S = \int_V(\delta\epsilon_{ss}\sigma_{ss} + \delta\epsilon_{st}\sigma_{st} + \delta\epsilon_{sn}\sigma_{sn} + \delta\epsilon_{tt}\sigma_{tt} + \delta\epsilon_{tn}\sigma_{tn})dV \tag{37}$$

in which the variational shell-surface strains are obtained by the following tensor transformation of the variational natural-coordinate strains:

$$\begin{bmatrix} \delta\epsilon_{ss} & \delta\epsilon_{st} & \delta\epsilon_{sn} \\ \delta\epsilon_{st} & \delta\epsilon_{tt} & \delta\epsilon_{tn} \\ \delta\epsilon_{sn} & \delta\epsilon_{tn} & 0 \end{bmatrix} = T_{sn}\begin{bmatrix} \delta\epsilon_{\zeta\zeta} & \delta\epsilon_{\zeta\eta} & \delta\epsilon_{\zeta\zeta} \\ \delta\epsilon_{\zeta\eta} & \delta\epsilon_{\eta\eta} & \delta\epsilon_{\eta\zeta} \\ \delta\epsilon_{\zeta\zeta} & \delta\epsilon_{\eta\zeta} & 0 \end{bmatrix}T^T_{sn} \tag{38}$$

These strains are then used to compute the shell-coordinate stress increments:

$$\Delta\sigma_s = [\,\Delta\sigma_{ss} \; \Delta\sigma_{st} \; \Delta\sigma_{sn} \; \Delta\sigma_{tt} \; \Delta\sigma_{tn}\,]^T \tag{39}$$

by adopting a suitable constitutive relation (e.g., Ref. 38)

$$\Delta\sigma_s = C \quad \Delta\epsilon_s \tag{40}$$

where

$$\Delta\epsilon_s = [\,\Delta\epsilon_{ss} \quad \Delta\epsilon_{st} \quad \Delta\epsilon_{sn} \quad \Delta\epsilon_{tt} \quad \Delta\epsilon_{tn}\,]^T \tag{41}$$

Once the stress increments are calculated, the total stresses are updated by:

$$\sigma^{(n+1)}_s = \sigma^{(n)}_s + \Delta\sigma^{(n+1)}_s \tag{42}$$

The incremental shell-surface, strain-displacement relations are obtained from the incremental, natural-coordinate, strain-displacement relations by the same tensor transformation. Specifically, for the present choice of the normal vector (Eq. 36) we have

$$T_{sn} = \begin{bmatrix} t_{s\zeta} & t_{s\eta} & 0 \\ t_{t\zeta} & t_{t\eta} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \frac{1}{|n_{\zeta}\times n_{\eta}|}\begin{bmatrix} s_{\xi}\cdot n_{\eta} & -s_{\xi}\cdot n_{\zeta} & 0 \\ -s_t\cdot n_{\eta} & s_t\cdot n_{\zeta} & 0 \\ 0 & 0 & |n_{\zeta}\times n_{\eta}| \end{bmatrix} \tag{43}$$

Expanding Eq. 38 while employing Eq. 43, we obtain the following explicit relation:

$$\begin{pmatrix} \delta\epsilon_{ss} \\ \delta\epsilon_{tt} \\ \delta\epsilon_{st} \\ \delta\epsilon_{sn} \\ \delta\epsilon_{tn} \end{pmatrix} = \begin{bmatrix} t^2_{s\zeta} & t^2_{s\eta} & t_{s\zeta}t_{s\eta} & 0 & 0 \\ t^2_{t\zeta} & t^2_{t\eta} & t_{t\zeta}t_{t\eta} & 0 & 0 \\ 2t_{s\zeta}t_{t\zeta} & 2t_{s\eta}t_{t\eta} & t_{s\zeta}t_{t\eta} + t_{s\eta}t_{t\zeta} & 0 & 0 \\ 0 & 0 & 0 & t_{s\zeta} & t_{s\eta} \\ 0 & 0 & 0 & t_{t\zeta} & t_{t\eta} \end{bmatrix}\begin{pmatrix} \delta\epsilon_{\zeta\zeta} \\ \delta\epsilon_{\eta\eta} \\ \delta\epsilon_{\zeta\eta} \\ \delta\epsilon_{\zeta\zeta} \\ \delta\epsilon_{\zeta\eta} \end{pmatrix} \tag{44}$$

Substitution of Eqs. 44 and 42 into Eq. 37, and evaluation of the resulting variational operator yields the desired internal force vector in terms of the shell-coordinate quantities. It should be noted that the strains, $\Delta\epsilon_{ij}$, are equivalent to classical strain-displacement relations for an orthogonal shell-surface coordinate system, with one major difference. In classical shell theories the unknowns are the covariant displacements ($\mathbf{a}\cdot\Delta\mathbf{u}$ and $\Delta\beta$), and their covariant derivatives. In the present formulation, the unknowns are the inertially-based displacements ($\Delta\mathbf{u}$ and $\Delta\hat{\mathbf{n}}$), and their covariant derivatives. This difference plays a crucial role in the finite element discretization of the resulting variational equations.

2.5 Resultant Form of Stiffness Force Operator

Using the strain-displacement relations in Eq. 44, the thin-shell counterpart to the stiffness force operator (Eq. 37) can be expressed in a resultant-force form:

$$\delta\mathcal{F}^S = \int_S \delta\mathfrak{e}_N^T\, \mathfrak{f}_N^S\, dS \tag{45}$$

and the strain and stress resultant "vectors"*, $\delta\mathfrak{e}_N$ and $\mathfrak{f}_N^S$, expressed in the natural (curvilinear) coordinate system, N, are defined as:

$$\delta\mathfrak{e}_N = \begin{Bmatrix} \delta\epsilon_{\alpha\alpha} \\ \delta\epsilon_{\beta\beta} \\ \delta\epsilon_{\alpha\beta} \\ \delta\kappa_{\alpha\alpha} \\ \delta\kappa_{\beta\beta} \\ \delta\kappa_{\alpha\beta} \\ \delta\gamma_{\beta n} \\ \delta\gamma_{\alpha n} \end{Bmatrix}, \quad \mathfrak{f}_N^S = \begin{Bmatrix} n^{\alpha\alpha} \\ n^{\beta\beta} \\ n^{\alpha\beta} \\ m^{\alpha\alpha} \\ m^{\beta\beta} \\ m^{\alpha\beta} \\ q^{\alpha} \\ q^{\beta} \end{Bmatrix} \tag{46}$$

While the covariant strain measures in $\delta\mathfrak{e}_N$ were defined in Eq. 44, the corresponding membrane, bending, and transverse-shear contravariant stress resultants, in $\mathfrak{f}_N^S$, are defined by pre-integration of Eq. 37 through the thickness, as follows:

$$n^{\alpha\beta} = \int_\zeta \sigma^{\alpha\beta}\, d\zeta\,, \qquad m^{\alpha\beta} = \int_\zeta \sigma^{\alpha\beta}\zeta\, d\zeta\,, \qquad q^{\alpha} = \int_\zeta \sigma^{t\alpha}\, d\zeta \tag{47}$$

where, again, α and β range from 1-2.

The last quantity to be explained in Eq. 45 is the differential reference surface area, dS. It arises by employing the thin-shell hypothesis to the volume integral in Eq. 37, i.e.,

$$\int_V (\,)\, dV \;=\; \int_\zeta \int_{S(\zeta)} (\,)\, dS(\zeta)\, d\zeta \;\simeq\; \int_S \int_\zeta (\,)\, d\zeta\, dS \tag{48}$$

3. ASSUMED NATURAL-COORDINATE STRAINS FOR 4-ANS ELEMENT

Originally a procedure for constructing shell elements whose strains were approximated along the natural-coordinate strain lines was presented in Refs. 28 and 38. We shall refer to it as the "old construction procedure." Our motivation for developing the old shell element procedure was to render the resulting shell element less sensitive to mesh distortions while maintaining the locking-free and mechanism-free features.

* We continue to use the term "vector" for one-dimensional arrays, but note that the components of vectors such as $\delta\mathfrak{e}_N$ and $\mathfrak{f}_N^S$ actually transform as second rank tensors.

In the old procedure, first, we projected the displacements and unit normals onto the natural-coordinate lines to obtain their covariant natural-coordinate quantities. Second, we interpolated the resulting covariant displacements and covariant unit normals. Third, we obtained the natural-coordinate derivatives of these covariant quantities along the element edges and the two natural-coordinate lines (or along the Hrennikoff lines). Finally, the strains in the element interior were obtained by interpolating the quantities along the Hrennikoff lines. It is important to note that the nodal displacements in the old construction were expressed in the covariant system, whereas they are expressed in the inertial system in the new element construction. This difference plays a key role in the subsequent element construction.

The first key feature of the present construction is the way we obtain the product form of the strain-displacement relations, viz, $\epsilon^e_{\xi\xi}$ along any ξ-line from Eq. 23:

$$\epsilon^e_{\xi\xi} = \mathbf{t}^T_\xi \cdot \frac{\partial \mathbf{u}}{\partial S_\xi} \tag{49}$$

in which the variational operator, δ, and the finite incremental operator, Δ, are omitted for presentation clarity, and $\mathbf{t}^T_\xi$ and $\frac{\partial \mathbf{u}}{\partial S_\xi}$ are recalled from Eqs. 10 and 32, respectively, as:

$$\mathbf{t}^T_\xi = \frac{1}{\lambda_\xi}(x_{,\xi} \quad y_{,\xi} \quad z_{,\xi}) \tag{50}$$

$$\frac{\partial \mathbf{u}}{\partial S_\xi} = \frac{1}{\lambda_\xi}\left(\frac{\partial u}{\partial \xi} \quad \frac{\partial v}{\partial \xi} \quad \frac{\partial w}{\partial \xi}\right)^T \tag{51}$$

Hence, in the new construction, first we interpolate the displacements fixed in the inertial coordinate system and the unit normals, $\mathbf{t}^T_\xi$, that vary along the ξ and η-lines. We then obtain their derivatives along the natural-coordinate lines. Third, we project the interpolated quantities and the derivatives onto the appropriate natural-coordinate lines to yield the necessary covariant derivatives. By combining them we obtain the desired covariant natural-coordinate strains.

The second key feature is the way in which we represent the natural-coordinate strains at any interior point of the element. In the old construction, we obtained the natural-coordinate strains along the Hrennikoff reference lines. The natural-coordinate strains at an element interior were then obtained by interpolating these reference-line strains. In the new construction, we do not use the Hrennikoff reference-line strains. Instead, at each integration point the necessary interpolations are performed along the two natural-coordinate lines passing through the integration point.

The third key feature — perhaps the most significant one — of the present procedure is the way in which the natural-coordinate inplane shear strain and twist, $\epsilon_{\xi\eta}$ and $\kappa_{\xi\eta}$, are interpolated. We abandoned the directionally reduced integration approach adopted in the old procedure. Instead, we sample these strains at the Barlow points, which are then tensorially transformed and interpolated at each integration point. We believe that the enhanced interpolations of $\epsilon_{\xi\eta}$ and $\kappa_{\xi\eta}$ are largely responsible for overcoming pathological 4-ANS element behavior for doubly curved shell surfaces. We describe the present construction of the assumed natural-coordinate strains for both 4-node and 9-node ANS shell elements.

We describe the interpolation procedures for the natural-coordinate strains (Eqs. 23-31) for the new 4-ANS shell element with the following interpolation functions (see Refs. 15 and 45 as regards the use of isoparametric shape functions):

$$
\begin{aligned}
NEN &= 4 \\
N_a(\xi,\eta) &= \sum_{r=1}^{2}\sum_{s=1}^{2} N_r(\xi)\,N_s(\eta) \\
N_1(\xi^a) &= \frac{1}{2}(1-\xi^a) \\
N_2(\xi^a) &= \frac{1}{2}(1+\xi^a) \\
a &= 2(s-1)+r \\
NB &= 1 \\
w_1(\xi^a) &= 1 \\
\xi_1^a &= 0.
\end{aligned}
\tag{52}
$$

3.1 Interpolations of $\epsilon_{\xi\xi}$, $\epsilon_{\eta\eta}$, $\kappa_{\eta\eta}$, and $\kappa_{\xi\xi}$

Let us recall from Eq. 23 the membrane strain expression:

$$
\epsilon^o_{\xi\xi} = t_\xi^T \frac{\partial u}{\partial S_\xi} \tag{23}
$$

Observe that along an η-constant line, $u_{,\xi}$ and $x_{,\xi}$ remain constant whereas, along a ξ-constant line, $u_{,\eta}$ and $x_{,\eta}$ remain constant. Therefore, A_ξ and t_ξ remain constant along any η-constant line whereas A_η and t_η remain constant along any ξ-constant line. These simple observations provide basic properties of the product form of the natural-coordinate strains for 4-node elements.

Hence, the axial membrane strain, $\epsilon^o_{\xi\xi}$, remains constant along any ξ-line, because t_ξ, A_ξ and $u_{,S_\xi}$ are constant along any ξ-line. Similarly, $\epsilon^o_{\eta\eta}$ remains constant along any η-line. Hence, the two membrane strains maintain a constant strain state along the two natural-coordinate lines, thereby satisfying the patch test requirement.

Interpolations of $\kappa_{\eta\eta}$ and $\kappa_{\xi\xi}$ are constructed in the same way as in the case of $\epsilon_{\xi\xi}$ and $\epsilon_{\eta\eta}$.

In other words, these four strains are obtained in a straightforward manner by substituting into Eqs. 23, 25, 28 and 31, the standard isoparametric coordinates and displacements, and their derivatives via the shape functions in Eq. 52.

3.2 Interpolation of Transverse Shear Strains, $\epsilon_{\xi z}$ and $\epsilon_{\eta z}$

Because there are several important features associated with the present interpolations of these two strains, we recall their expression from Eqs. 26 and 27:

$$
\gamma_\xi = t_\xi^T \frac{\partial u}{\partial S_\xi} + t_\xi^T \hat{u} \tag{26}
$$

$$
\gamma_\eta = t_\eta^T \frac{\partial u}{\partial S_\eta} + t_\eta^T \hat{u} \tag{27}
$$

First, let us address the well-known transverse shear locking problem (e.g., Refs. 46, 31, 11 and 20): that is, the interpolation of the second terms, $t_\xi^T\hat{u}$ and $t_\eta^T\hat{u}$, in the above two transverse shear strains. In the present construction, for the term, $t_\xi^T\hat{u}$, to be constant along the ξ-line, we must have $\hat{u}$ constant because t_ξ^T remains constant along the ξ-line. This can be accomplished by adopting the following interpolation for $\hat{u}$:

$$
\hat{u} = \sum_{r=1}^{2}\sum_{s=1}^{2} N_r(0)\,N_s(\eta)\cdot\hat{u}_{(r,s)} \tag{53}
$$

Similarly, we adopt for the interpolation of $t_\eta^T\hat{u}$:

$$
\hat{u} = \sum_{r=1}^{2}\sum_{s=1}^{2} N_r(\xi)\,N_s(0)\cdot\hat{u}_{(r,s)} \tag{54}
$$

so that it remains constant along the η-line.

The net effect of the above two modifications is the same as the widely adopted reduced integration procedure used to avoid the transverse shear locking phenonemon discussed in Refs. 11, 20, 10, 33, 12, 44, 3, 23, 26 and 27. However, no rank deficiency is introduced as a consequence of the present modifications in $\hat{u}$ for both γ_ξ and γ_η. This is because the η-dependency in $\hat{u}$ for γ_ξ and the ξ-dependency in $\hat{u}$ for γ_η are *not* compromised as a result of the foregoing modifications. Further discussions on rank deficiency vs $(\xi,\ \eta)$-dependency are in Ref. 23.

Second, we will address the first term in γ_ξ, viz, $t_\xi^T \frac{\partial u}{\partial S_\xi}$. Because we must have, for consistency, a constant value for this term along the ξ-line, the unit normal, t_ξ^T, must remain constant along the ξ-line.

Observe, if t_ξ^T is to remain constant along the ξ-line, $\{x_{,\eta}\ \ y_{,\eta}\ ,\ \ z_{,\eta}\}$ and A_η must be evaluated at $\xi = 0$ for each integration point while still substituting the appropriate value for η. To be specific, we evaluate:

$$
t_\xi^T(\xi = 0,\ \eta_j)\cdot\frac{\partial u}{\partial S_\xi}(\xi_i,\ \eta_j) \tag{55}
$$

where the coordinates, $(\xi_i,\ \eta_j)$, denote spatial (2×2)-integration points.

For γ_η, we evaluate in the opposite way, viz,

$$
t_\eta^T(\xi_i,\ \eta = 0)\cdot\frac{\partial u}{\partial S_\eta}(\xi_i,\ \eta_j) \tag{56}
$$

3.3 Interpolations of $\epsilon_{\xi\eta}$ and $\kappa_{\xi\eta}$

The derivatives in the inplane shear strain, $\epsilon^o_{\xi\eta}$, and the twist, $\kappa_{\xi\eta}$, must be evaluated to preserve constant strain states along each of the natural-coordinate lines. It is noted that $\epsilon^o_{\xi\eta}$ consists of two product terms: $t_\xi^T\cdot\frac{\partial u}{\partial S_\eta}$ and $t_\eta^T\cdot\frac{\partial u}{\partial S_\xi}$. The first term implies that the parameteric derivative, $\frac{\partial u}{\partial S_\eta}$, is projected on the vector component, t_ξ, that is parallel to the ξ-line. In the first term, $\frac{\partial u}{\partial S_\eta}$ remains a constant along η-line. Hence, if the first term is to maintain a constant strain state, so must t_ξ.

To maintain such constant strain states with minimum mesh sensitivity, we introduce the following approximations. We sample $\epsilon_{\xi_o\xi_o}$, $\epsilon_{\xi_o\eta_o}$ and $\epsilon_{\eta_o\eta_o}$ at the element centroid ξ_o, η_o. At each integration point ξ, η, we introduce a rotation vector, n, such that:

$$
n = (n_{\xi_o}\times n_\xi)/|n_{\xi_o}\times n_\xi| \tag{57}
$$

which rotates the normal vector n_{ξ_o} at the centroid to coincide with the normal vector n_ξ at the integration point. The projection of n_{ξ_o}, n_{η_o} at the centroid onto the shell surface at the integration point is thus obtained (Ref. 8):

$$
n_{\xi_o'} = (n_{\xi_o}\cdot n_\xi)n_{\xi_o} + (1-n_{\xi_o}\cdot n_\xi)(n\cdot n_{\xi_o})n - (n_\xi\cdot n_{\xi_o})n_{\xi_o} \tag{58}
$$

$$
n_{\eta_o'} = (n_{\xi_o}\cdot n_\xi)n_{\eta_o} + (1-n_{\xi_o}\cdot n_\xi)(n\cdot n_{\eta_o})n - (n_\xi\cdot n_{\eta_o})n_{\xi_o} \tag{59}
$$

where $n_{\xi_o'}$, $n_{\eta_o'}$ represent the projection of n_{ξ_o}, n_{η_o} on the shell surface at the integration point ξ, η.

Hence, from Eqs. 57 - 59 and 14 one obtains:

$$\begin{Bmatrix} a_\xi \\ a_\eta \end{Bmatrix} = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{Bmatrix} a_{\xi_o} \\ a_{\eta_o} \end{Bmatrix} \tag{60}$$

The inplane shear strain at the integration point is obtained via the following tensorial transformation:

$$\epsilon_{\xi\eta} = 2t_{11}t_{21}\epsilon_{\xi_o\xi_o} + 2t_{12}t_{22}\epsilon_{\eta_o\eta_o} + (t_{11}t_{22} + t_{12}t_{21})\epsilon_{\xi_o\eta_o} \tag{61}$$

The projected inplane strain, $\epsilon_{\xi\eta}$ given by Eq. 61, thus maintains a constant strain state at each integration point.

When the shell surface is flat, the preceding treatment of the inplane strain interpolation appears to be akin to the transformation of the tensor stress τ^{ij} to the physical stress component σ^{ij} for which Pian *et al.* used the isoparametric Jacobian matrix evaluated at the element centroid (Refs. 32 and 17). For the finite element discretization based on the natural-coordinate system, the present interpolation of $\epsilon_{\xi\eta}$ as given by Eq. 61 and the assumed stress interpolation of $\sigma^{\xi\eta}$ evaluated at the centroid in Ref. 32 are different. The present interpolation replaces the inplane shear strain at each integration point with the one at the centroid, via the centroid-to-integration point shell surface coordinate transformation. On the other hand, a straightforward application of the procedure of Ref. 32 would require that the strains at each integration point are first transformed to the corresponding ones in terms of the centroid natural-coordinate system. The transformed inplane bending component at the integration point is then replaced by the centroidal one. Both approaches can be easily implemented for 4-node elements. However, for nine-node elements, extension of the present interpolation for nine-node becomes easier as discussed in the following section.

4. STRAIN INTERPOLATION FOR 9-ANS ELEMENT

When the shell surfaces are approximated by the isoparametric curved shape functions (Ref. 1) the limitation principle of Ref. 7 states that the strains should vary linearly for nine-node elements. From the theoretical viewpoints, the mitigation of element locking and spurious mechanisms resulting from reduced integration can be considered as efforts to adhere to Fræjis de Veubeke's limitation principle. Efforts to mitigate both locking and spurious mechanisms for nine-node elements can be found in Refs. 1, 5, 6, 14, 16, 21, 22, 28, 38, and 44.

In essence, the new 9-ANS shell element is based on the independent approximations of the two fields in terms of the nodal variables: the displacement and the strain field within an element interior. The choice of the *nodal* displacements and rotations as the nodal variables for the present 9-ANS shell element presents only a special case; they can be stresses or strains, as well. Thus, this philosophy is adopted in the discrete (finite element) version of the thin shell equations for a nine-node shell element.

However, while those two fields are interdependent (through the strain-displacement relations), the assumed strain approach takes the liberty of selecting the approximations independently — each in the nodal displacement variables. This is similar to what is done for "hybrid" elements via a mixed variational principle, except that there an element-level matrix inversion is required to achieve the linkage between strains and nodal displacements, while in the assumed-strain approach this linkage is made explicit.

This allows most of the usual element requirements (e.g., continuity of the displacement field, completeness of the strain field, convergence, locking-free behavior, etc.) to be met a priori. However, in particular, the assumed natural-coordinate strain (ANS) approach focuses on the physical covariant components of the strain field to reduce element sensitivity to mesh distortion. We will now describe the construction of a new 9-ANS shell element in detail.

4.1 Interpolation of $\epsilon^o_{\xi\xi}$

In the old construction, the curvilinear membrane strain along the ξ-line for a fixed η-line employed the formula:

$$\bar{\epsilon}^o_{\xi\xi} = \frac{\partial u_\xi}{\partial S_\xi} = \frac{1}{A_\xi}\frac{\partial u_\xi}{\partial \xi} \tag{62}$$

wherein u_ξ is the covariant displacement defined by:

$$u_\xi = \frac{\xi}{2}(\xi - 1)\bar{u}_1 + (1 - \xi^2)\bar{u}_2 + \frac{\xi}{2}(\xi + 1)\bar{u}_3 \tag{63}$$

in which $\bar{u}_i$ are the covariant components at the nodal points, i, that are tangent along the ξ-line. Substituting Eq. 63 into Eq. 62 and using the relations Eq. 13, one can derive an explicit form for $\bar{\epsilon}^o_{\xi\xi}$. It was shown in Ref. 25 that the strain $\bar{\epsilon}^o_{\xi\xi}$, thus derived, introduces inconsistencies. The complicated interpolations offered therein can be viewed as corrective measures to improve $\bar{\epsilon}^o_{\xi\xi}$.

To illustrate the present membrane-strain construction, let us consider a nine-node quadratic shell surface for which both its coordinates and the displacements are interpolated by the bi-quadratic Lagrange shape functions as given by:

$$\begin{aligned}
NEN &= 9 \\
N_a(\xi,\eta) &= \sum_{r=1}^{3}\sum_{s=1}^{3} N_r(\xi)\,N_s(\eta) \\
N_1(\xi^\alpha) &= \frac{1}{2}\xi^\alpha(\xi^\alpha - 1) \\
N_2(\xi^\alpha) &= (1 - (\xi^\alpha)^2) \\
N_3(\xi^\alpha) &= \frac{1}{2}\xi^\alpha(\xi^\alpha + 1) \\
a &= 3(s-1) + r \\
NB &= 2 \\
w_1(\xi^\alpha) &= \frac{1}{2}(1 - \sqrt{3}\xi^\alpha) \\
w_2(\xi^\alpha) &= \frac{1}{2}(1 + \sqrt{3}\xi^\alpha) \\
\xi^\alpha_1 &= -1/\sqrt{3} \\
\xi^\alpha_2 &= +1/\sqrt{3}
\end{aligned} \tag{64}$$

The most one can realize for the covariant natural-coordinate strain, $\epsilon^o_{\xi\xi}$, when interpolated by Eq. 62, is a linearly varying field along the ξ-line. In Ref. 21 the strains based on the quadratic polynomials yield an equivalent of the desired linearly varying field only if sampled at the two Barlow points (Ref. 2), $\xi_b = \pm 1/\sqrt{3}$. Hence, a linearly varying strain field along the ξ-coordinate line can be constructed by the following strain interpolation:

$$\epsilon^o_{\xi\xi} = \frac{1}{2}(\epsilon^o_{\xi\xi}(\xi_b) + \epsilon^o_{\xi\xi}(-\xi_b)) + \frac{\xi}{2\xi_b}(\epsilon^o_{\xi\xi}(\xi_b) - \epsilon^o_{\xi\xi}(-\xi_b)), \qquad \xi_b = 1/\sqrt{3} \tag{65}$$

which was extensively used in Ref. 28.

We now come to the second key aspect of the present construction: that is, the way we obtain the natural-coordinate strains at any point in the element interior. Notice that the covariant membrane strain, $\epsilon_{\xi\xi}$, given by Eq. 65 can directly represent the natural-coordinate strain at any point in the element because the directional derivatives with respect to ξ, i.e.,

ϵ_ξ and u_ξ, represent its values at any η-line. This is an important improvement over the old construction wherein the strains in the element interior were obtained by interpolating the strains along the six Hrennikoff lines.

For computer implementation case, we re-express Eq. 65 in the following form:

$$\epsilon^*_{\xi\xi} = \sum_{a=1}^{NEN} b^{\xi\xi}_{u_a} u^a \tag{66}$$

where:

$$b^{\xi\xi}_{u_a} = \sum_{i=1}^{NB} w_i(\xi) \left[N_{a,s} t^T_\xi(\xi) \right]_{\xi_i,\eta} \tag{67}$$

where $w_i(\xi)$ are the weighting functions as give in Eq. 64.

The above explicit form for $\epsilon^*_{\xi\xi}$ constitutes a key contribution of the present 0-ANS construction as it is valid everywhere in the element; it is linearly varying along the ξ-line. Hence, we circumvented the strain interpolation of $\epsilon^*_{\xi\xi}$ by using the strains along the three Hrennikoff lines, viz, $\eta = (-1, 0, +1)$, as previously used in Ref. 28.

The strain interpolation along the ξ-line for $\epsilon_{\xi\xi}$, as derived in Eq. 65, does not require reduced integration as it exactly satisfies the constant and linearly varying strain states, thus causing no element locking. A similar approach was adopted in Refs. 4, 41, 42 and 10. This locking-free property of the present element construction that is distinct from the family of curved shell elements based on the standard isoparametric construction. In addition, as long as one invokes nine-point integration, the resulting element possesses its full rank, thus no spurious mechanism occurs. A symbolic analysis illuminating this characteristic is in Ref. 23.

In the assumed nonphysical covariant strain approach (see Ref. 16), one interpolates a nonphysical strain:

$$\epsilon^{cov}_{\xi\xi} = t^T_\xi \cdot \frac{\partial u_\xi}{\partial \xi} \tag{68}$$

Hence, the difference between the present physical-component strain given by Eq. 62 and the non-physical covariant strain is the absence of A_ξ in its denominator. If $\epsilon^{cov}_{\xi\xi}$ is interpolated to vary linearly along the ξ-line, then the two formulations coincide only if A_ξ is constant along the ξ-line. This happens only for a constant curvature, viz, when the ξ-lines lie on a circle. For distorted meshes, even though the elements may lie on a cylinder or sphere, the ξ-line does not necessarily lie on a constant curvature trajectory. This difference may play a crucial role on element performance for distorted meshes.

Interpolations of $\epsilon_{\eta\eta}$, $\kappa_{\xi\xi}$ and $\kappa_{\eta\eta}$ follow a similar procedure for interpolating $\epsilon_{\xi\xi}$ as described above.

4.2 Interpolation of Transverse Shear Strains, $\epsilon_{\xi\zeta}$ and $\epsilon_{\eta\zeta}$

We recall again the two strains from Eqs. 26 and 27:

$$\gamma_\xi = t^T_\xi \frac{\partial u}{\partial S_\xi} + t^T_\xi \hat{u} \tag{26}$$

$$\gamma_\eta = t^T_\xi \frac{\partial u}{\partial S_\eta} + t^T_\eta \hat{u} \tag{27}$$

To avoid locking because of inconsistent interpolations of the transverse shear strains, one must employ the same interpolation procedure adopted for $\epsilon_{\xi\xi}$ in the preceding section. The resulting expression for the two interpolated transverse shear strains become:

$$\gamma_\xi = \frac{1}{2}(\gamma_\xi(\xi_b) + \gamma_\xi(-\xi_b)) + \frac{\xi}{2\xi_b}(\gamma_\xi(\xi_b) - \gamma_\xi(-\xi_b)), \qquad \xi_b = 1/\sqrt{3} \tag{69}$$

$$\gamma_\eta = \frac{1}{2}(\gamma_\eta(\xi_b) + \gamma_\eta(-\xi_b)) + \frac{\xi}{2\xi_b}(\gamma_\eta(\xi_b) - \gamma_\eta(-\xi_b)), \qquad \xi_b = 1/\sqrt{3} \tag{70}$$

For computer implementation case, we re-express the above equations in the following form:

$$\gamma_\xi = \sum_{a=1}^{NEN} b^{\xi\zeta}_{u_a} u^a + \sum_{a=1}^{NEN} b^{\xi\zeta}_{\theta_a} \hat{u}^a \tag{71}$$

where

$$b^{\xi\zeta}_{u_a} = \sum_{i=1}^{NB} w_i(\xi) \left[N_{a,s} t^T_\xi \right]_{\xi_i,\eta} \tag{72}$$

$$b^{\xi\zeta}_{\theta_a} = \sum_{i=1}^{NB} w_i(\xi) \left[N_a t^T_\xi \right]_{\xi_i,\eta} \tag{73}$$

Similarly, an explicit implementable expression for γ_η can be obtained by interchanging (ξ_i, η) with (ξ, η_i) into the preceding equation.

4.3 Interpolations of $\epsilon_{\xi\eta}$ and $\kappa_{\xi\eta}$

Essentially, we extend the procedure outlined for the 4-ANS case to the 0-ANS element as follows. We sample $\epsilon_{\xi\xi}$, $\epsilon_{\xi\eta}$ and $\epsilon_{\eta\eta}$ at the four Barlow points, $(\xi = \pm 1/\sqrt{3}, \eta = \pm 1/\sqrt{3})$. At each integration point (ξ, η), we transform the four inplane shear strains evaluated at the four Barlow points via the same tensorial transformation done for the 4-ANS case (Eq. 61):

$$\epsilon^b_{\xi\eta} = 2t^b_{11}t^b_{21}\epsilon_{\xi_b\xi_b} + 2t^b_{12}t^b_{22}\epsilon_{\eta_b\eta_b} + (t^b_{11}t^b_{22} + t^b_{12}t^b_{21})\epsilon_{\xi_b\eta_b} \tag{74}$$

where the superscript and the subscript, b, refers to one of the four Barlow points and the matrix components, t^b_{ij}, have the same geometrical meaning as the one in Eq. 60, except they relate the projection on the shell surface from a Barlow point to the integration point.

The inplane shear strain, $\epsilon_{\xi\eta}$, at the integration point is then obtained by interpolating $\epsilon^b_{\xi\eta}$ as follows:

$$\epsilon_{\xi\eta} = \sum_{b=1}^{4} W_b(\xi,\eta) \cdot \epsilon^b_{\xi\eta} \tag{75}$$

where

$$\boxed{\begin{aligned} W_a(\xi,\eta) &= \sum_{r=1}^{2}\sum_{s=1}^{2} w_r(\xi)\, w_s(\eta) \\ w_1(\xi^n) &= \frac{1}{2}(1 - \sqrt{3}\xi^n) \\ w_2(\xi^n) &= \frac{1}{2}(1 + \sqrt{3}\xi^n) \\ a &= 2(s-1) + r \end{aligned}} \tag{76}$$

Similarly, the twist, $\kappa_{\xi\eta}$, is obtained by interchaning $\epsilon^b_{\xi\eta}$.

5. EVALUATION OF PRESENT 0-ANS SHELL ELEMENTS

We present a theoretical analysis of the present 0-ANS element for an inextensional bending case and for a cylinder subjected to uniform pressure. We will then give a preliminary performance of the present 4-ANS and 0-ANS elements for two simple shell problems: a pinched cylinder and a pinched sphere.

5.1 Inextensional Bending of Arch

The linearly varying strains derived in the preceding section should yield a locking-free and mechanism-free curved shell element for most applications. For thin shells, however, an accurate solution to inextensional bending problems remains an important part of shell analysis for applications such as sheet metal forming. For an arch in Fig. 2, the present strain-displacement interpolations for the membrane strain (Eq. 65), the bending strain that is obtained by replacing u, v, w, in Eq. 65 with $\hat{u}$, $\hat{v}$, $\hat{w}$, and the transverse shear strain as given by Eq. 69, respectively, yield the following results:

$$\epsilon^{\circ}_{\xi\xi} = \frac{1}{A_1^2}\{-R\sin\phi\, u_0 + \tfrac{4}{3}R(1-\cos\phi)\, w_0\} \tag{77}$$

$$\kappa_{\xi\xi} = \frac{\frac{\phi}{\sin\phi}\cdot\left\{1+\frac{8}{3}\left(\frac{1-\cos\phi}{\sin\phi}\right)^2\right\}}{\left\{1+\frac{4}{3}\left(\frac{1-\cos\phi}{\sin\phi}\right)^2\right\}\cdot\left\{1+4\left(\frac{1-\cos\phi}{\sin\phi}\right)^2\right\}^{\frac{1}{2}}}\cdot\left(\frac{1}{R'}-\frac{1}{R}\right) \tag{78}$$

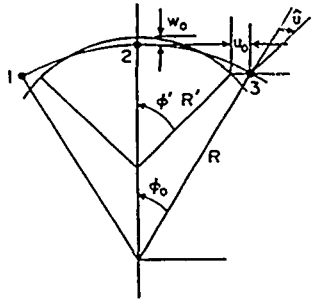
$$\gamma_\xi = \frac{\xi R\sin\phi}{2A_1}(\phi'-\phi)f(\phi) \quad , \tag{79}$$

where

$$f(\phi) = 1-\left[1+\tfrac{4}{3}\left(\frac{1-\cos\phi}{\sin\phi}\right)^2\right]^{\frac{1}{2}}\frac{2(1-\cos\phi)}{\phi\sin\phi}\frac{1}{1+\frac{1}{3}(1-\cos\phi)} \tag{80}$$

The theoretical solutions to the above inextensional bending case are:

$$\epsilon^{\circ}_{\xi\xi}=0 \quad , \quad \kappa_{\xi\xi}=\left(\frac{1}{R'}-\frac{1}{R}\right) \quad , \quad \gamma_\xi=0 \tag{81}$$



ARCH LENGTH, $S = 2R\phi = 2R'\phi'$

ROTATION, $\hat{u} = \phi' - \phi = \left(\frac{1}{R'}-\frac{1}{R}\right)\frac{S}{2}$

Fig. 2 - Inextensional Deformation of an Arch

It is easily verified that the membrane strain in Eq. 77 remains zero because of the small strain-increment assumption invoked in the present formulation. Hence, the two remaining measures of accuracy for the present strain approximations for this inextensional bending case are: the error in the curvature itself, $e_{bending}$, and the error in the transverse shear relative to the curvature change, e_{shear}:

$$e_{bending} = 1-\frac{\frac{\phi}{\sin\phi}\cdot\left\{1+\frac{8}{3}\left(\frac{1-\cos\phi}{\sin\phi}\right)^2\right\}}{\left\{1+\frac{4}{3}\left(\frac{1-\cos\phi}{\sin\phi}\right)^2\right\}\cdot\left\{1+4\left(\frac{1-\cos\phi}{\sin\phi}\right)^2\right\}^{\frac{1}{2}}} \tag{82}$$

$$e_{shear} \approx O\left[\frac{\int\gamma_\xi^2 d\xi}{\int(\zeta\kappa_{\xi\xi})^2 d\zeta}\right]^{\frac{1}{2}} \simeq \frac{R}{h}f(\phi) \tag{83}$$

Fig. 3 shows the errors in the present approximation of the bending strain in terms of the element size, ϕ. For $\phi = 60°$ which corresponds to the element arc length of $2R\phi$, the error remains within one percent. In Fig. 4, the errors in the relative transverse shear as defined in Eq. 83 are plotted against the shell parameter R/h for one-tenth of a percent and one percent error. Only for extremely thin shells, for example, $R/h \sim 1000$, which is not shown in the figure, one needs $\phi \leq 20°$ if the ratio in the two energy components, e_{shear}, is to remain less than one percent. Hence, we conclude that the present formulation can capture inextensional bending deformations with adequate accuracy, provided the element size is not too large.

5.2 Cylinder Under Uniform Internal Pressure

For the case of a cylinder subjected to uniform internal pressure, we obtain from Eqs. 19 and 24:

$$\epsilon^{\circ}_{\xi\xi} = \frac{w_0}{R} \quad , \quad \gamma_\xi = 0 \tag{84}$$

which is exact. Hence, the present formulation yields the exact solution for this classical problem.

5.3 Preliminary Numerical Results

The present four and nine-node ANS-shell elements were implemented and tested for two simple shell cases, viz., cylinder under a concentrated load and a pinched hemisphere. The performance of both of the new 4-ANS and 9-ANS elements for the pinched cylinder problem, although not reported here, indicates that the new 9-ANS element (designated as 9-rANS) overshoots the solution. For the 4-ANS element, both the old and new 4-ANS manifest about the same convergence rate. This, however, is not the case for the hemisphere problem. As shown in Fig. 5, the present 9-ANS shell element (designated as 9-rANS) converges at the second grid (9×9) whereas the old 9-ANS element converges at the refined grid (17×17). As for the present 4-ANS element, the new element maintains the same performance as the one obtained by the old 4-ANS element with a rigid-mode projection. This is a significant improvement over the old 4-ANS element and the 4-STG/P element that is also compared in Refs. 25 and 38.

A significant change in the construction of the new family ANS elements has been in the way the inplane bending strain and twist ($\epsilon_{\xi\eta}, \kappa_{\xi\eta}$) are interpolated, even though interpolations of the rest of the strain components have been somewhat improved from the old construction. We will report in the near future on the improved performance of the new 4-ANS and 9-ANS elements in production-level shell analysis.

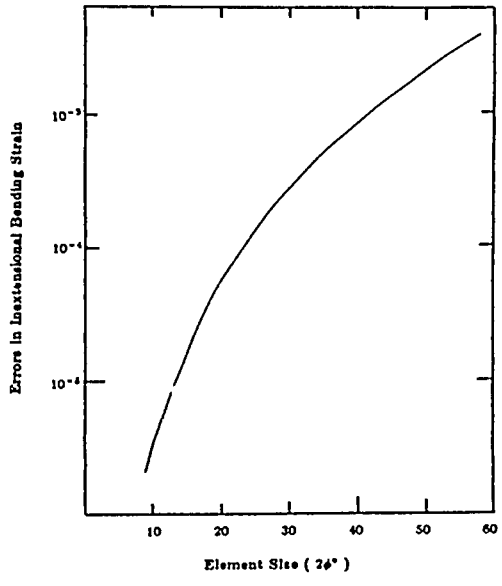


Fig. 3 - Errors in the Bending Strain for Inextensional Bending of an Arch

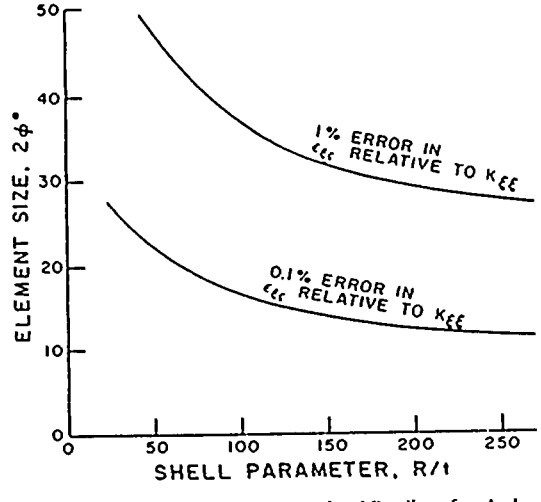


Fig. 4 Errors in Transverse Shear for Inextensional Bending of an Arch

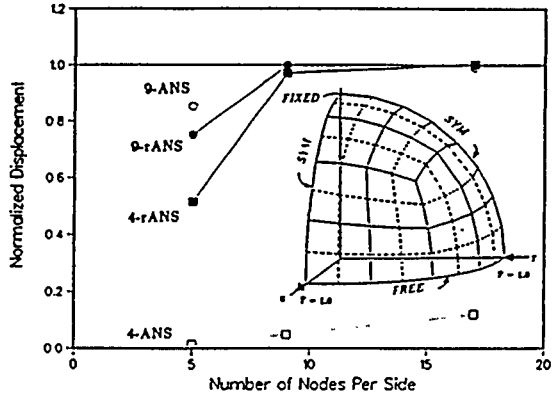


Fig. 5 - Pinched Hemisphere Element Convergence Study: Normalized Center Displacement vs. Grid Density

6. DISCUSSION

We have presented in detail the construction of both 4-ANS and 9-ANS shell elements based on the formulation in Ref. 25 that is suitable for assumed natural strain shell elements. The major emphasis of that formulation has been to incorporate as much shell behavior as possible into the basic formulation. One important consequence of this emphasis is the modification of the inplane shear and twist. Other possibilities exist in the basic formulation, which may be further improved to better capture shell behavior, because a complete hierarchical approximation of the basic formulation has not yet been carried out.

While we have endeavored to preserve the well-known *first-order* thin shell theory with transverse shear effects, the present element construction has avoided two related difficulties stemming from a direct application of the classical thin shell equations: rigid-body motions and the derivatives of the two fundamental surface coefficients, A_ξ and A_η. Avoidance of these difficulties by the present element construction was accomplished by use of the inertially fixed translational displacements and the corotationally based, two-rotational variables.

Even though we construct the basic element attributes, the strain-displacement matrix (known as B-matrix), based on the natural-coordinate system, we proceed with the evaluation of the internal stiffness force based on the orthogonal shell-surface coordinates. Hence, the present element can be easily *plugged into* any existing shell analyzer for both geometrical and material nonlinear problems. This is in contrast with the so-called "covariant shell elements" of Refs. 5 and 16, wherein one works with a set of non-physical strains, which amounts to embedding the fundamental shell-surface coefficients into their corresponding constitutive matrix. For example, years of experience in the solution procedures for plasticity analysis based essentially on the amount of physical strain increments may not be of use in the solution of plasticity problems based on the covariant strain elements, because the amount of covariant strain increments is depends on the element size.

Although not elaborated in this paper, the strain increments need not be infinitesimal. In particular, the displacement and rotation increments for large rigid motions can be arbitrarily large. Specifically, for nonlinear elasticity problems without bifurcation possibility, the present

element together with an element-independent corotational algorithm (Ref. 34) need not accumulate stresses, because the displacement and rotation increments can be measured from the initial state to the current state.

An error analysis of the present element for uniform membrane strain state and inextensional bending state (Fig. 2) illustrates that the present 9-ANS element possesses improved transverse shear modeling and membrane modeling compared with the old 9-ANS element. For example, in no case does the error in the computed inextensional bending exceed one percent, for up to a 90^0-span. This is reported in Fig. 3. The mean square error in the transverse shear energy, though, restricts the allowable element size as the shell becomes thinner, as illustrated in Fig. 4. A preliminary numerical test of the new 4-ANS and 9-ANS shell elements on the pinched cylinder and pinched hemisphere indicates that the new elements improve significantly for doubly curved shell surfaces. We will examine in more detail their potential improvements through production level computations.

As for improving the accuracy of the transverse shear strains, this is where a rigorous three-dimensional analysis should shed light on the reliability of C^0-type thin shell elements. We intend to study this aspect in the near future. It should be noted, however, that such errors are consistent within the bounds of errors in most thin shell theories.

ACKNOWLEDGEMENTS

The work reported herein was partially supported by the Naval Research Laboratory under Contract N00014-87-K-2118 and by the Independent Research Program of Lockheed Missiles and Space Co. We thank Ms. Louise Schuets of NRL for her interest and encouragement.

REFERENCES

1. Ahmad, S., Irons, B. M. and Zienkiewicz, O. C., "Analysis of Thick and Thin Shell Structures by Curved Elements," International Journal for Numerical Methods in Engineering, Vol. 2, 1970, pp. 419-451.

2. Barlow, J., "Optimal Stress Locations in Finite Element Models," International Journal for Numerical Methods in ' 'eering, Vol. 10, 1976, pp. 243-251.

3. Crisfield, M. A., "A Four-Noded Thin-Plate Element Using Shear Constraints- A Modified Version of Lyons' Element," Computer Methods in Applied Mechanics and Engineering, Vol. 38, 1983, pp. 93-120.

4. Crisfield, M. A., "A Quadratic Mindlin Element Using Shear Constraints," Computer Methods in Applied Mechanics and Engineering, Vol. 18, 1984, pp. 833-852.

5. Dvorkin, E. N. and Bathe, K. J., "A Continuum Mechanics Based Four-Node Shell Element for General Nonlinear Analysis," International Journal for Computer-Aided Engr. & Software, Vol. 1, 1984, pp. 77-88.

6. Ergatoudis, I., Irons, B. M. and Zienkiewicz, O C., "Curved Isoparametric Quadrilateral Elements for Finite Element Analysis," International Journal of Solids and Structures, Vol. 4, 1968, pp. 31-42.

7. Fræijs de Veubeke, B , "Displacement and Equilibrium Models in the Finite Element Method", in Stress Analysis, O. C. Zienkiewicz and G. Holister, eds., John Wiley, 1965, pp. 145-197.

8. Gibbs, J. W., Vector Analysis, Dover, New York, N. Y., 1960, pp. 368-371.

9. Hrennikoff, A., "Solution of Problems of Elasticity by the Framework Method," Journal of Applied Mechanics, Vol. 8, 1941, pp. A169-175.

10. Huang, H.C. and Hinton, E., "A New Nine Node Degenerated Shell Element with Enhanced Membrane and Shear Interpolation," International Journal for Numerical Methods in Engineering, Vol. 22, 1986, pp. 73-92.

11. Hughes, T. J. R., Taylor, R. C. and Kanoknukulchai, W., "A Simple and Efficient Finite Element for Plate Bending," International Journal for Numerical Methods in Engineering, Vol. 11, 1977, pp. 1529-1547.

12. Hughes, T. J. R. and Tezduyar, T. E., "Finite Elements Based on Mindlin Plate Theory with Particular Reference to the Four-Node Bilinear Isoparametric Elements," Journal of Applied Mechanics, Vol. 48, 1981, pp. 587-596.

13. Irons, B., "Engineering Application of Numerical Integration in Stiffness Methods," AIAA J., Vol. 4, 1966, pp. 2035-2037.

14. Irons, B. M., "The Semiloof Shell Element," in Finite Elements for Thin Shells and Curved Members, D. G. Ashwell and R. H. Gallagher, eds., John Wiley, London, 1976, pp. 197-222.

15. Irons, B. and Ahmad, S., Techniques for Finite Elements, John Wiley, New York, 1980.

16. Jang, J, "A Nine Node Shell Element Based On Assumed Covariant Strain Interpolation," PhD Dissertation, Stanford University, Stanford, California, 1987.

17. Kang, D. S. and Pian, T. H. H., "A Versatile and Low Order Hybrid Stress Element for General Shell Geometry," Proceedings of the 28th Structures, Structural Dynamics and Materials Conference, 1987, pp. 633-641.

18. Knight, N. F., McCleary, S. L., Macy, S. C. and Aminpour, M. A., "Large-Scale Structural Analysis: The Structural Analyst, The CSM Testbed and The NAS System," NASA Technical Memorandum 100643, NASA Langley Research Center, Hampton, Va., March 1989.

19. Lee, S. W. and Pian, T. H. H., "Improvement of Plate and Shell Finite Elements by Mixed Formulations," AIAA J., Vol. 16, 1978, pp. 29-34.

20. MacNeal, R. H., "A Simple Quadrilateral Shell Element," Computers & Structures, Vol. 8, 1978, pp. 175-183.

21. MacNeal, R. H., "Derivation of Element Stiffness Matrices by Assumed Strain Distribution," Nuclear Engineering Design, Vol. 70, 1982, pp. 3-12.

22. Parisch, H., "A Critical Survey of the Nine-Node Degenerated Shell Element with special Emphasis on Thin Shell Application and Reduced Integration," Computer Methods in Applied Mechanics and Engineering, Vol. 20, 1979, pp. 323-350.

23. Park, K. C., "Symbolic Fourier Analysis Procedures for C^0 Finite Elements," in Innovative Methods for Nonlinear Analysis, W. K. Liu, T. Belytschko and K. C. Park, eds., Pineridge Press, Swansea, 1984, pp. 269-293.

24. Park, K. C., "An Improved Strain Interpolation for Curved C^0 Elements," the Bruce Irons Memorial Issue of International Journal for Numerical Methods in Engineering, Vol. 22, 1986, pp. 281-288.

25. Park, K. C., "The ANS Shell Elements: Part I – Formulation," CU-CSSC-87-06, Center for Space Structures and Controls, University of Colorado, Boulder, CO, 1987.

26. Park, K. C. and Flaggs, D. L., "A Symbolic Fourier Synthesis of a One-Point Integrated Quadrilateral Plate Element," Computer Methods in Applied Mechanics and Engineering, Vol. 48, 1985, pp. 203-236.

7. Park, K. C., Stanley, G. M. and Flaggs, D. L., "A Uniformly Reduced, Four-Noded C^0-Shell Element with Consistent Rank Corrections," Computers & Structures, Vol. 20, 1985, pp. 129-139.

8. Park, K. C. and Stanley, G. M., "A Curved C^0 Shell Element Based on Assumed Natural-Coordinate Strains," Journal of Applied Mechanics, Vol. 108, 1986, pp. 278-290.

9. Park, K. C. and Stanley, G. M. and Cabiness, H., "A Family of C^0 Shell Elements Based on Generalized Hrennikoff's Method and Assumed Natural-Coordinate Strains," in Finite Element Methods for Nonlinear Problems, Bergan, P. et al, eds., Springer-Verlag, Berlin, 1986, pp. 265-282.

0. Park, K. C., Stanley, G. M., "Strain Interpolations for A Four-Node ANS Shell Element," Computational Mechanics '88, S. N. Atluri and G. Yagawa (eds.), Vol. 1, Springer-Verlag, New York, 1988, pp. 26.i.1-4.

1. Pawsey, S. F. and Clough, R. W., "Improved Numerical Integration of Thick Shell Finite Elements," International Journal for Numerical Methods in Engineering, Vol. 3, 1971, pp. 575-586.

2. Pian, T. H. H. and Sumihara, K., "Rational Approach for Assumed Stress Finite Elements," International Journal for Numerical Methods in Engineering, Vol. 20, 1984, pp. 1685-1695.

3. Pugh, E. D. L., Hinton, E. and Zienkiewicz, O. C., "A Study of Quadrilateral Plate Bending Elements with Reduced Integration," International Journal for Numerical Methods in Engineering, Vol. 12, 1978, pp. 1059-1079.

4. Rankin, C. C. and Brogan, F. A., "An Element-Independent Corotational Procedure for the Treatment of Large Rotations," Journal of Pressure Vessel Technology, Vol. 108, 1986, pp. 165-174.

5. Reissner, E., "The Effect of Transverse Shear Deformation on the Bending of Elastic Plates," Journal of Applied Mechanics, 1945, pp. A69-A77.

6. Sanders, J. L, "An Improved First Approximation Theory for Thin Shells," NASA-TR-R24, 1959.

7. Salmon, D. C., "Large Change-of-Curvature Effects in Quadratic Finite Elements for CAD of Membrane Structures," PhD Dissertation, Cornell University, Ithaca, New York, 1987.

8. Stanley, G. M., "Continuum-Based Shell Analysis," PhD Dissertation, Stanford University, Stanford, California, 1985.

9. Stanley, G. M., Park, K. C. and Hughes, T. J. R., "Continuum-Based Resultant Shell Elements," Finite Element Method for Plate and Shell Structures, Volume 1: Element Technology, Hughes, T. J. R. and Hinton, E., eds., Pineridge Press, Swansea, U. K., 1986, pp. 1-45.

10. Stanley, G. M., Cabiness, H. and Park, K. C., "Revised ANS Shell Elements: Implementation and Numerical Evluations," Computational Mechanics '88, S. N. Atluri and G. Yagawa (eds.), Vol. 1, Springer-Verlag, New York, 1988, pp. 26.v.1-4.

11. Stolarski, H. and Belytschko, T., "Membrane Locking and Reduced Integration for Curved Elements," Journal of Applied Mechanics, Vol. 49, 1982, pp. 172-176.

42. Stolarski, H. and Belytschko, T., "Shear and Membrane Locking in Curved C^0 Elements," Computer Methods in Applied Mechanics and Engineering, Vol. 41, 1983, pp. 279-296.

43. Wempner, G. A., Oden, T. J. and Kross, D. A., "Finite Element Analysis of Thin Shells," Journal of Engineering Mechanics, Division, ASCE, Vol. 49, 1968, pp. 1273-1294.

44. Wempner, G., Talaslidis, D. and Huang, C. - M., "A Simple and Efficient Approximation of Shells Via Finite Quadrilateral Elements," Journal of Applied Mechanics, Vol. 49, 1982, pp. 115-120.

45. Zienkiewicz, O. C., The Finite Element Method in Engineering Science, McGraw-Hill, New York 1971.

46. Zienkiewicz, O. C., Taylor, R. C. and Too, J. M., "Reduced Integration Technique in General Analysis of Plates and Shells," International Journal for Numerical Methods in Engineering, Vol. 3, 1971, pp. 275-290.

Revised ANS Shell Elements: Implementation and Numerical Evaluation

Gary Stanley*, Harold Cabiness* and K.C. Park†

*Computational Mechanics Section, Lockheed Palo Alto Research Laboratories Palo Alto, CA 94304, USA
†Department of Aerospace Engineering Sciences & Center for Space Structures & Controls, University of Colorado, Boulder, CO 80309, USA

Summary

This paper looks at some of the fine points and implementation aspects of a revised formulation for assumed natural-coordinate strain (ANS) shell elements, which has been developed by Park, Stanley and Cabiness [1]. After reviewing the essential features of the formulation, various "implementation options" are described and evaluated numerically. The experience gained here should be applicable to other shell element formulations.

1. Introduction

Assumed natural-coordinate strain (ANS) shell elements have been presented in references [1-4]. Related elements have appeared earlier (see, e.g., [5]) and have been appearing with increasing frequency since (see, e.g., [6-8]). In all of these cases, the primary goal has been to reduce the sensitivity of quadrilateral shell elements to mesh distortion — both in-plane and out-of-plane (warping). While considerable improvement has been obtained with respect to the fundamental isoparametric formulation (with or without selective/reduced integration), there is the nagging feeling that more robustness is still available from ANS-type shell elements. The purpose of the present paper is to examine several aspects of the improved ANS shell element reported in [1] that were not made explicit in the formulation, and hence were relegated to the whim of the implementor. After describing the basic ANS formulation, these implementation "options" are delineated and their effect studied numerically.

2. Basic ANS Formulation

The linearized strain components used in the basic ANS shell element formulation [2] are simply the covariant (natural-coordinate) components of the continuum strains

$$\delta\epsilon = [\nabla\delta\mathbf{u}]^{sym} \tag{1}$$

where, $\delta\mathbf{u}$ is the linearized displacement vector, subject to the Reissner/Mindlin-type shell constraint of straight normals, i.e.,

$$\begin{aligned} \mathbf{x}(\xi,\eta,\zeta) &= \bar{\mathbf{x}}(\xi,\eta) + \zeta\hat{\mathbf{x}}(\xi,\eta) \\ \delta\mathbf{u}(\xi,\eta,\zeta) &= \delta\bar{\mathbf{u}}(\xi,\eta) + \zeta\delta\hat{\mathbf{u}}(\xi,\eta) \end{aligned} \tag{2}$$

in which $\mathbf{x}$ is the position vector of an arbitrary point in the shell, $\bar{\mathbf{x}}$ points to its "anchor" on the reference surface, and $\hat{\mathbf{x}}$ is the unit normal vector along the line connecting the two points. Similarly, $\delta\bar{\mathbf{u}}$ and $\delta\hat{\mathbf{u}}$ are the translational and rotational (relative) displacement vectors associated with the reference surface and the unit normal, respectively. In these expressions, ξ,η are the non-dimensional (natural) surface

coordinates, and ζ is the physical thickness coordinate for the shell. With this constraint, the linearized natural-coordinate shell strain measures may be expressed (to first order in ζ) as

$$\delta\epsilon_{\alpha\beta}(\xi,\eta,\zeta) = \delta\bar{\epsilon}_{\alpha\beta}(\xi,\eta) + \zeta\delta\kappa_{\alpha\beta}(\xi,\eta), \qquad \delta\epsilon_{\zeta\alpha}(\xi,\eta,\zeta) = 2\,\delta\gamma_{\alpha}(\xi,\eta) \tag{3}$$

where α and β take on values 1, 2, designating ξ,η components, respectively. In particular, the *membrane* strains are defined as:

$$\delta\bar{\epsilon}_{\xi} = \bar{\mathbf{a}}_{\xi}\cdot\frac{\partial\delta\bar{\mathbf{u}}}{\partial\xi}, \quad \delta\bar{\epsilon}_{\eta} = \bar{\mathbf{a}}_{\eta}\cdot\frac{\partial\delta\bar{\mathbf{u}}}{\partial\eta}, \quad \delta\bar{\epsilon}_{\xi\eta} = \bar{\mathbf{a}}_{\xi}\cdot\frac{\partial\delta\bar{\mathbf{u}}}{\partial\eta} + \bar{\mathbf{a}}_{\eta}\cdot\frac{\partial\delta\bar{\mathbf{u}}}{\partial\xi} \tag{4}$$

the *bending* strains (changes-in-curvature) are defined as:

$$\delta\kappa_{\xi} = \bar{\mathbf{a}}_{\xi}\cdot\frac{\partial\delta\hat{\mathbf{u}}}{\partial\xi} + \delta\kappa_{\xi}^{cor}, \quad \delta\kappa_{\eta} = \bar{\mathbf{a}}_{\eta}\cdot\frac{\partial\delta\hat{\mathbf{u}}}{\partial\eta} + \delta\kappa_{\eta}^{cor}, \quad \delta\kappa_{\xi\eta} = \left(\bar{\mathbf{a}}_{\xi}\cdot\frac{\partial\delta\hat{\mathbf{u}}}{\partial\eta} + \bar{\mathbf{a}}_{\eta}\cdot\frac{\partial\delta\hat{\mathbf{u}}}{\partial\xi}\right) + \delta\kappa_{\xi\eta}^{cor} \tag{5}$$

and the *transverse-shear* strains are defined as:

$$\delta\gamma_{\xi} = \hat{\mathbf{x}}\cdot\frac{\partial\delta\bar{\mathbf{u}}}{\partial\xi} + \bar{\mathbf{a}}_{\xi}\cdot\delta\hat{\mathbf{u}}, \quad \delta\gamma_{\eta} = \hat{\mathbf{x}}\cdot\frac{\partial\delta\bar{\mathbf{u}}}{\partial\eta} + \bar{\mathbf{a}}_{\eta}\cdot\delta\hat{\mathbf{u}} \tag{6}$$

In (4)-(6), $\bar{\mathbf{a}}_{\xi}$ and $\bar{\mathbf{a}}_{\eta}$ are the covariant basis vectors tangent to ξ and η on the reference surface, i.e.,

$$\bar{\mathbf{a}}_{\xi} = \frac{\partial\bar{\mathbf{x}}}{\partial\xi}, \qquad \bar{\mathbf{a}}_{\eta} = \frac{\partial\bar{\mathbf{x}}}{\partial\xi} \tag{7}$$

and the *cor* superscript represents curvature correction terms to be discussed below.

Next, the ANS formulation approximates (discretizes) these strains by expressing them directly in terms of element nodal displacement variables. Briefly, in order to obtain rank sufficiency, no locking, and minimal sensitivity to mesh distortion, each natural-coordinate strain component is assumed to vary as a prescribed polynomial function of ξ and η — by extrapolating from a set of *isoparametrically* based strains which are evaluated at special (reduced-integration or Barlow) sampling points. Thus, for example, the membrane strain, $\delta\bar{\epsilon}_{\xi\xi}$, and the transverse-shear strain, $\delta\gamma_{\eta}$, are assumed as follows

$$\delta\bar{\epsilon}_{\xi\xi}(\xi,\eta) \approx \sum_{i=1}^{NB}\rho_i(\xi)\,\delta\bar{\epsilon}_{\xi\xi}^{iso}(\xi_i,\eta) = \sum_{i=1}^{NB}\rho_i(\xi)\left[\bar{\mathbf{a}}_{\xi}(\xi_i,\eta)\cdot\sum_{a=1}^{NEN}\frac{\partial N_a}{\partial\xi}(\xi_i,\eta)\,\delta\bar{\mathbf{u}}_a\right] \tag{8}$$

$$\begin{aligned}\delta\gamma_{\eta}(\xi,\eta) \approx \sum_{j=1}^{NB}\rho_j(\eta)\,\delta\gamma_{\eta}^{iso}(\xi,\eta_j) = \sum_{j=1}^{NB}\rho_j(\eta)\Bigg[&\hat{\mathbf{x}}(\xi,\eta_j)\cdot\sum_{a=1}^{NEN}\frac{\partial N_a}{\partial\xi}(\xi,\eta_j)\,\delta\bar{\mathbf{u}}_a \\ &+ \bar{\mathbf{a}}_{\eta}(\xi,\eta_j)\cdot\sum_{a=1}^{NEN}N_a(\xi,\eta_j)\,\delta\hat{\mathbf{u}}_a\Bigg]\end{aligned} \tag{9}$$

where the *iso* superscript indicates that the standard isoparametric (Lagrange) shape functions, $N_a(\xi,\eta)$, are employed, $\delta\bar{\mathbf{u}}_a$ and $\delta\hat{\mathbf{u}}_a$ are the translation and rotation vectors at element node a, and ρ_i and ξ_i,η_i are weighting functions and Barlow point coordinates that depend on the number of element nodes (e.g., 4 or 9).

3. Implementation Options

The ambiguity of the above formulation (and others like it) arises in at least two ways: (i) choice of covariant strain components: physical versus tensor, and (ii) choice of element geometric parameters: i.e., the means of constructing the covariant tangent and normal basis vectors that appear in the definition of strain. While it is not evident from the basic formulation, differences in these choices can have a significant effect on element performance. The key options are as follows.

Physical versus Tensor Components

If physical components of strain are chosen instead of the tensor components used above, then the covariant basis vectors, $\bar{\mathbf{a}}_\xi$ and $\bar{\mathbf{a}}_\eta$, appearing in (4)–(6) become unit vectors, $\hat{\mathbf{a}}_\xi$ and $\hat{\mathbf{a}}_\eta$, and the derivatives with respect to ξ and η require scaling by $1/\|\bar{\mathbf{a}}_\xi\|$ and $1/\|\mathbf{a}_\eta\|$, respectively. More interestingly, however, is that the curvature corrections in (5) take on a substantially different form, depending on whether physical or tensor components are employed. For example, the twisting curvature correction, $\delta\kappa^{cor}_{\xi\eta}$, in terms of tensor components is simply

$$\delta\kappa^{cor}_{\xi\eta} = \frac{\partial\hat{\mathbf{x}}}{\partial\xi}\cdot\frac{\partial\delta\bar{\mathbf{u}}}{\partial\eta} + \frac{\partial\hat{\mathbf{x}}}{\partial\eta}\cdot\frac{\partial\delta\bar{\mathbf{u}}}{\partial\xi} \tag{10}$$

while in terms of physical components it becomes:

$$\begin{aligned}\delta\kappa^{cor}_{\xi\eta\,P} &= \Big[\frac{\partial\hat{\mathbf{x}}}{\partial\xi_P} - \big[(\hat{\mathbf{a}}_\xi\cdot\frac{\partial\hat{\mathbf{x}}}{\partial\xi_P}) + (\hat{\mathbf{a}}_\eta\cdot\frac{\partial\hat{\mathbf{x}}}{\partial\eta_P})\big]\hat{\mathbf{a}}_\xi\Big]\cdot\frac{\partial\delta\bar{\mathbf{u}}}{\partial\eta_P} \\ &+ \Big[\frac{\partial\hat{\mathbf{x}}}{\partial\eta_P} - \big[(\hat{\mathbf{a}}_\xi\cdot\frac{\partial\hat{\mathbf{x}}}{\partial\xi_P}) + (\hat{\mathbf{a}}_\eta\cdot\frac{\partial\hat{\mathbf{x}}}{\partial\eta_P})\big]\hat{\mathbf{a}}_\eta\Big]\cdot\frac{\partial\delta\bar{\mathbf{u}}}{\partial\xi_P}\end{aligned} \tag{11}$$

where $\partial(\,)/\partial\alpha_P = (\partial(\,)/\partial\alpha)/\|\bar{\mathbf{a}}_\alpha\|$. While the above two expressions are equivalent in a continuum sense, differences arise due to discretization.

Choice of Element Geometric Parameters

There are at least three different ways to compute the covariant basis vectors, $\bar{\mathbf{a}}_\xi, \bar{\mathbf{a}}_\eta$, and the unit normal vector, $\hat{\mathbf{x}}$, at element integration points — as required in the discrete strain-displacement relations, e.g., (8)–(9). First, one can use a *strict isoparametric* interpretation, in which

$$\bar{\mathbf{a}}_\xi(\xi,\eta) \approx \sum_{a=1}^{NEN}\frac{\partial N_a}{\partial\xi}\bar{\mathbf{x}}_a, \qquad \bar{\mathbf{a}}_\eta(\xi,\eta) \approx \sum_{a=1}^{NEN}\frac{\partial N_a}{\partial\eta}\bar{\mathbf{x}}_a, \qquad \hat{\mathbf{x}}(\xi,\eta) \approx \sum_{a=1}^{NEN} N_a\hat{\mathbf{x}}_a \tag{12}$$

Alternatively, one can use the isoparametric recipe for the tangent vectors, but a *modified approach for the unit normal* vectors, i.e.

$$\hat{\mathbf{x}}(\xi,\eta) \approx \bar{\mathbf{a}}_\xi \times \bar{\mathbf{a}}_\eta/\|\bar{\mathbf{a}}_\xi \times \bar{\mathbf{a}}_\eta\| \tag{13}$$

Finally, one can use a modified approach for *both* the tangent and normal vectors, by interpolating the tangent vectors via:

$$\bar{\mathbf{a}}_\xi \approx \sum_{a=1}^{NEN}\bar{\mathbf{a}}^a_\xi, \qquad \bar{\mathbf{a}}_\eta \approx \sum_{a=1}^{NEN}\bar{\mathbf{a}}^a_\eta, \tag{14}$$

and computing the normal vectors via (13). Additionally, if relations (12) or (14) are used, there is the also the choice of how to compute the nodal normal or tangent vectors. Here, one can either use the *element* geometry, (as defined by the nodal coordinates, $\bar{\mathbf{x}}_a$, and the shape functions N_a), or the *exact* geometry (e.g., as defined by a CAD system).

While it can be shown that the first option is the only one that preserves rigid-body invariance (i.e., zero straining), it is not necessarily the best option for general-purpose use.

4. Numerical Evaluation

Numerical results comparing the effects of the above "implementation options" are now being gathered, and will be presented in person at the conference. Additional work is proceeding to predict these effects *symbolically*, so that the best options can be designed into the formulation at the outset — rather than later, by trial and error.

Acknowlegements

The work reported herein was partially supported by the Naval Research Laboratory, contract N00014-87-K-2118, via a subcontract from the University of Colorado, Boulder; all numerical experiments were performed using the Computational Structural Mechanics (CSM) software Testbed, which is being jointly developed with NASA Langley Research Center under contract NAS1-18444.

References

1. Park, K.C. and Stanley, G.M., 1988, "Strain Interpolations for a 4-Node ANS Shell Element", ICESS 88 – Atlanta.

2. Park, K.C. and Stanley, G.M., 1986, "A Curved C^0 Shell Element Based on Assumed Natural-Coordinate Strains", *Journal of Applied Mechanics*, **108**, 278–290.

3. Park, K.C., 1986, "An Improved Strain Interpolation for Curved C^0 Elements", *International Journal for Numerical Methods in Engineering*, **22**, 281–282.

4. Park, K.C. and Stanley, G.M., 1987, "ANS Shell Element; Part I – Formulation", Center for Space Structures and Controls, University of Colorado, Report CU-CSSC-87-06.

5. MacNeal, R.H., 1982, "Derivation of Element Stiffness Matrices by Assumed Strain Distribution", *Nuclear Engineering Design*, **70**, 3–12.

6. Jang, J.H. and Pinsky, P., 1987, "A Nine-Node Assumed Covariant Strain Shell Element", to appear in International Journal for Numerical Methods in Engineering.

7. Huang, H.C. and Hinton, E., 1984, "A Nine-Node Lagrangian Plate Element with Enhanced Shear Interpolation", *Engineering Computations*, **1**, 369–379.

8. Stanley, G.M., Park, K.C., and Hughes, T.J.R., 1986, "Continuum-Based Resultant Shell Elements", in *Finite Element Methods for Plate and Shell Structures, Vol. 1: Formulations and Algorithms*, eds. T.J.R. Hughes and E. Hinton.

9. Kang, D.S. and Pian, T.H.H., 1987, "A Versatile and Low Order Hybrid Stress Element for General Shell Geometry", *AIAA*, **87-0840**, 633–641.